THE UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

# COMPOSITIONAL FLOW IN POROUS MEDIA: A RESERVOIR SIMULATOR

A DISSERTATION

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

BY

FORREST F. CRAIG, III

Norman, Oklahoma

1976

COMPOSITIONAL FLOW IN POROUS MEDIA:

A RESERVOIR SIMULATOR

APPROVED BY:

DISSERTATION COMMITTEE

## ACKNOWLEDGEMENTS

I wish to express my sincere appreciation to the Chemical Engineering and Materials Science Department and the Petroleum and Geological Engineering Department of the University of Oklahoma for their efforts and encouragement in this work. Thanks is particularly due Dr. Henry B. Crichlow for his help and supervision during the course of this study. Appreciation is also extended to the other members of the supervisory committee, Dr. K. E. Starling, Dr. D. E. Menzie, Dr. J. H. Christensen, and Dr. M. E. Chenevert for their valuable suggestions in the final preparation of this dissertation.

Financial support by the Petroleum and Geological Engineering Department, the Chemical Engineering and Materials Science Department, and the U.S. Department of Health, Education, and Welfare is gratefully acknowledged.

Special appreciation and love is extended to my wife, Shelley for her love, understanding, and support during the difficult times, and to my parents, Forrest F. and Elizabeth J. Craig for their encouragement. Without the support and encouragement of these people and organizations, this work would not have been possible.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

## LIST OF ILLUSTRATIONS

# COMPOSITIONAL FLOW IN POROUS MEDIA:

# A RESERVOIR SIMULATOR

## INTRODUCTION

> . . .[The phase behavior portion] of the problem will be treated by making the conventional assumption that "gas is gas", although this must be recognized as a basic weakness.
> (Muskat, [1], p. 381)

Muskat was only one of the many early investigators to recognize that the phase behavior aspects of the reservoir engineering analysis have traditionally been greatly oversimplified. Advances over the past decade or so in computing machinery and numerical techniques now present a means of treating the general reservoir problem, and specifically the hydrocarbon phase behavior problem, in a more rigorous manner.

This investigation attempts to formulate a general phase behavior algorithm consistent with the fundamentals of phase behavior which can be easily modified to describe observed fluid and reservoir behavior. This algorithm will be incorporated in a numerical reservoir simulation model. The main emphasis of this work will be to produce an efficient, flexible, and reasonably useful reservoir simulation package, concentrating on the development of the phase behavior algorithms.

# CHAPTER I

## HISTORY OF RESERVOIR SIMULATION

The engineer is constantly attempting to describe physical systems in mathematical terms. This effort to simulate the behavior of physical systems begins by making a great many assumptions to simplify the problem to a level which can be easily handled. As expertise in a given area increases, fewer simplifying assumptions are made and the solutions obtained begin to match the physical problem more closely. At each stage of improvement in the model, the engineer who uses it must be aware of the assumptions involved in order to fully understand the results obtained.

The same method of development has occurred in reservoir simulation. The initial effort to model a reservoir system was the zero-dimension material balance method. This procedure was based on a great many assumptions but provided the engineer with an indication of the expected behavior of a reservoir system. The natural next step in the advancement of simulation was the expansion of the zero-dimensional model to a one-dimensional scheme. This procedure has the advantages of fewer assumptions, greater flexibility, and as a result, greater applications and efficiency. In order to

obtain these advantages, the amount of computation is increased and there are a great many additional variables which must be included to describe the physical behavior in each cell of the system.

With the increasing availability of computers and their greater storage and faster computation speeds, the two-dimensional system was used to describe reservoir systems even more accurately. Although three-dimensional systems are now available, the most popular simulator is the two-dimensional procedure because a well-organized two-dimensional simulator can be used to approximate three-dimensional reservoir behavior. This results in a savings in execution time and storage requirements as a result of fewer cells, which is desired from an economic viewpoint. Two-dimensional models can be run on smaller computer systems and at a lower price than a three-dimensional model of the same system. The additional accuracy of a three-dimensional model is seldom justified due to the inaccuracies of the data input. No matter how well a model obeys the equations which describe real behavior, it is only as accurate as the information which is available on the reservoir properties.

Along with the advent of simulators which describe reservoir systems in multi-dimensional form, an effort was made to duplicate the behavior of the fluids present. The traditional approach (often called the black-oil method) was to view the fluid as several independent phases which had no interaction. As expertise increased in fluid behavior and

mathematical techniques of analysis, efforts were made to incorporate more accuracy into simulators. This was done in many ways, but the most popular was to use a table of lookup or polynomial curve fit of the fluid properties which required a minimal amount of additional storage. This resulted in a more realistic model, but this approach to the fluid properties assumed that they were functions of only pressure in an isothermal system, that interphase mass transfer did not occur, and that the phase composition did not change. The next advance was to use a compositional approach in which the fluid properties are described as functions of pressure, temperature, and composition of the fluids. A basic assumption involved in the compositional approach is that the water phase exists only as a liquid, while the hydrocarbon portion of the fluid can exist in either a gas or liquid phase. This approach requires the storage of not only the fluid properties for each cell, but also the vapor and liquid phase compositions (one variable for each component in each cell). The determination of the phase compositions for each cell requires increased computer time compared with the black-oil approach. Again, the model is only as good as the input data and the accuracy of the equations used.

The primary effort in this study is to develop a two-dimensional, compositional reservoir simulation package. Special attention will be paid to the fluid properties portion, particularly to the flash determination of the phase compositions. The use of compatible fluid property correlations for

the liquid and vapor phases is essential in the compositional approach. The correlations must produce a continuous function, especially at the critical point of the hydrocarbon mixture where the vapor and liquid properties are required to become indistinguishable. Minimization of computer storage and execution time is also a concern in order to provide an economical simulation package. Because compositional packages are, by nature, lengthy projects which require many man-years to optimize, this study will lay the foundation for a long term program. All of the components needed in a compositional simulator will be developed, any problem areas pinpointed, and recommendations will be made on a possible plan of research to improve these areas.

# CHAPTER II

## MECHANICS OF COMPOSITIONAL FLOW

A compositional model is one in which the phase behavior of the fluids in the reservoir is taken into account. It differs from a traditional black-oil model in that interphase mass transfer is considered and the individual components are treated, along with pressures and phase saturation, as separate variables. In order to understand the development and advantages of a compositional reservoir model, one must first understand the mechanics of compositional flow.

### Three-Phase Flow in Porous Media

Nearly all naturally occurring petroleum reservoirs consist of water-wet rock systems. These reservoirs, sometimes called hydrophilic reservoirs, exhibit the following distribution when a water-oil-gas system is present:

Water fills the smallest pores completely and wets the surface of all of the rock grains.

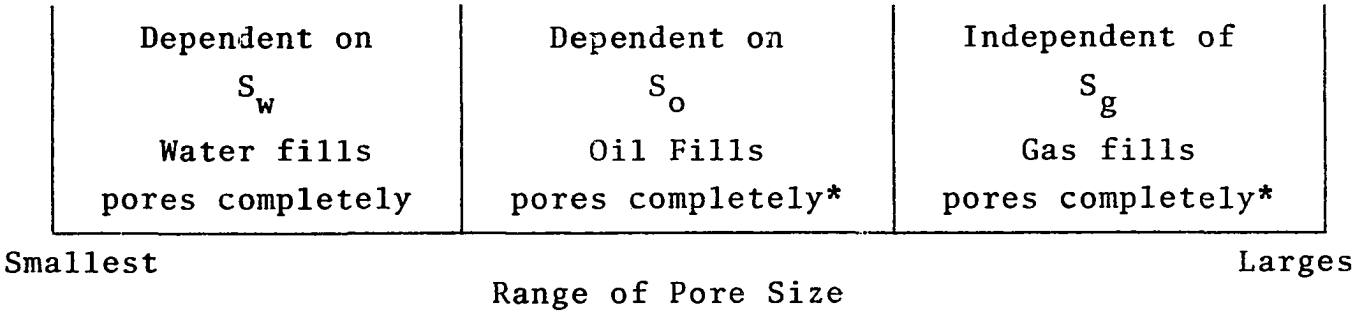
Oil fills a range of pores beginning with those which just exceed the maximum size completely filled with water. The amount of oil in the system determines the largest pore which is filled by the oil phase.

Gas fills the remainder of the pores, from those which just exceed the maximum size completely filled with oil to the largest pores present in the system. Because of the highly compressible nature of gases, this phase will expand to fill all of the pores not occupied by one of the other phases.

This behavior is shown diagrammatically in Figure 1. As would be expected, the pore sizes in naturally occurring rock are randomly distributed throughout the system. Macroscopically, the three phases appear to be well mixed because of the random pore distribution. There is ample opportunity under flow conditions for the phases to contact each other and for phase changes to occur.

## Classification of Reservoir Fluids

Each hydrocarbon system has its own phase diagram. A typical phase diagram, which depends only on the fluid composition, is presented in Figure 2. The critical point is that point where the liquid and vapor phases become indistinguishable. The composition, density, viscosity, and all other properties of one phase become identical to the corresponding property of the other phase at the critical point. The cricondentherm is defined as the maximum temperature at which a two-phase region can exist for a given fluid. The two-phase region is that area of the pressure-temperature phase diagram which is between the bubble point and dew point curves. This is the region where the reservoir fluid exists in equilibrium as both a gas and a liquid. Outside this




*except for surface of rock grains which are wet by water in hydrophilic systems.

Figure 1

Fluid Distribution in Pores

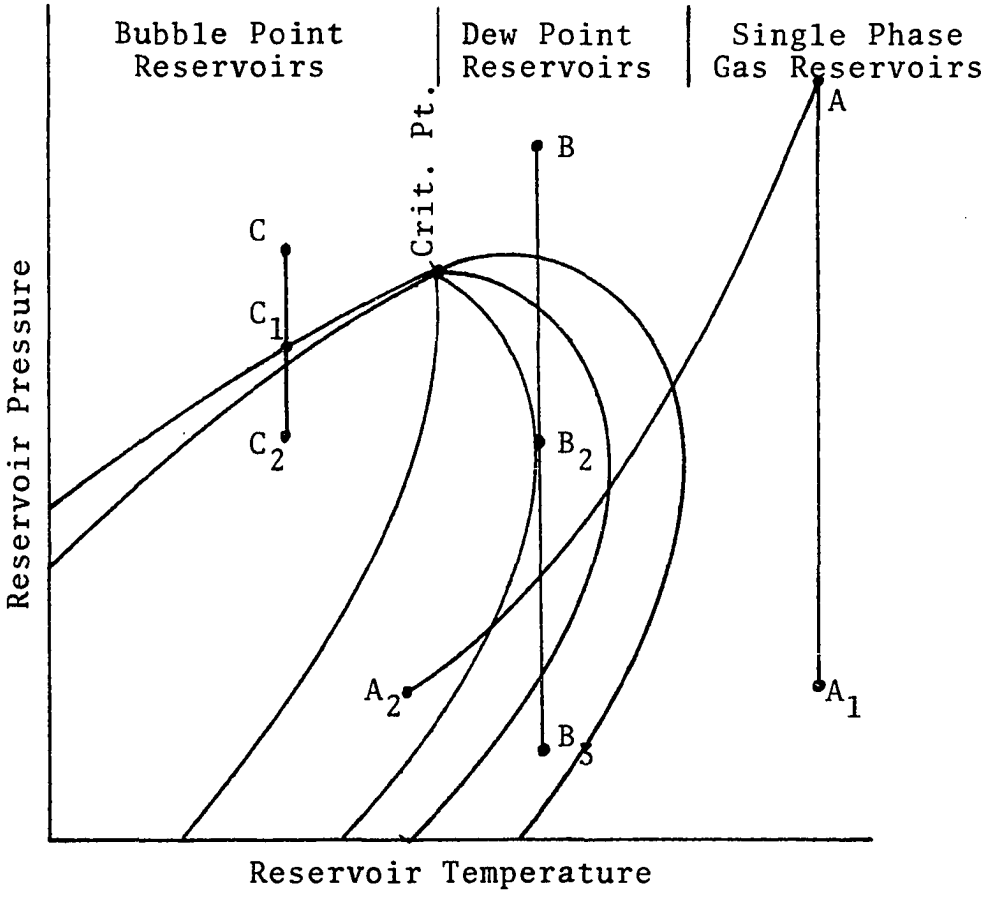



Figure 2

Pressure-Temperature Phase Diagram for
a Multicomponent Hydrocarbon System

region, the fluid exists only in one phase (liquid or gas). Reservoirs and the fluids they contain are classified based on the location on the phase diagram of the point which represents the initial temperature and pressure of the hydrocarbon mixture with respect to the critical point, the cricondentherm, and the two-phase region.

For a reservoir fluid which exists at the initial pressure and temperature conditions of point A on the phase diagram, we observe fluid behavior which follows the path from A to $A_1$ as the reservoir is depleted isothermally. Because the temperature of the reservoir exceeds the cricondentherm, the fluid in the reservoir will remain in a single (vapor) phase. This type of reservoir is called a single-phase gas system. During production, the fluid temperature and pressure decrease as the hydrocarbon mixture moves toward the surface. If the path of production enters the two-phase region, as in this case (path A - $A_2$), a liquid phase will condense out of the vapor and be produced from surface separators, although the fluid in the reservoir will continue to be single phase.

If the reservoir temperature lies between the critical temperature and the cricondentherm of the given hydrocarbon system and in the one-phase region, isothermal depletion of this gas system follows the path B - $B_3$. This system is called a retrograde gas-condensate or dew-point reservoir. As the reservoir pressure decreases, the phase path intersects the dew point curve. At this point ($B_1$) liquid begins

to form and the liquid volume present increases to a maximum at point $B_2$. From this point to $B_3$, the amount of liquid present decreases until abandonment pressure. Of course, if the abandonment pressure causes the path to cross the lower portion of the dew point curve, the reservoir fluid will again be a single (vapor) phase. It should be pointed out that the liquid phase which condenses forms on the pore surfaces of the rock and has a very low mobility. Thus, the composition of the reservoir fluid is constantly changing as the light hydrocarbon components are produced in the vapor phase while the heavy components condense out in a mostly unrecoverable liquid phase. This means the pressure-temperature phase diagram is constantly shifting downward and to the right (as the reservoir fluid composition changes), which further aggravates the retrograde liquid loss. If the reservoir pressure is maintained by natural water influx or gas cycling, this retrograde loss is reduced.

A bubble-point reservoir is characterized by an initial temperature less than the system critical temperature and an initial pressure greater than the bubble point pressure. Because of the low compressibility of liquids, the pressure decreases rapidly along the path C - $C_1$, as the reservoir liquid is produced. At point $C_1$, a vapor phase begins to form. Because this gas phase has a much higher mobility than the liquid phase, the gas-oil ratio for the produced fluid increases rapidly. When an active water drive is present, the reservoir pressure is at least partially

maintained. Because the reservoir pressure remains above the bubble pressure for a longer period of time, the single phase liquid flow into the well is maintained longer, so the oil recovery is higher. As in a retrograde reservoir, the vaporization of the light components and the high gas-oil ratio causes the composition of the reservoir fluid to shift continually. The phase diagram shifts to the right and downward which accentuates the decrease in the liquid recovery. In those cases where the shift is serious, the bubble-point reservoir is also called a volatile oil reservoir.

The fourth and final type of reservoir is the two-phase system. This type of reservoir is one with an initial temperature and pressure which falls inside the two-phase envelope of the pressure-temperature phase diagram. The gas cap phase produces as a retrograde or single-phase reservoir, while the liquid phase produces as a bubble-point reservoir. Each phase produces as a separate type of fluid because the vapor and liquid each have different compositions and thus different phase diagrams.

The four types of reservoir fluids are presented in Figure 3. The single-phase gas type is indicated by a reservoir temperature greater than the cricondentherm of the system, while a gas condensate reservoir is present when the reservoir temperature is less than the cricondentherm but greater than the critical temperature. The volatile oil and dark oil types of systems are different sub-types of bubble-point systems (a reservoir temperature less than the system critical

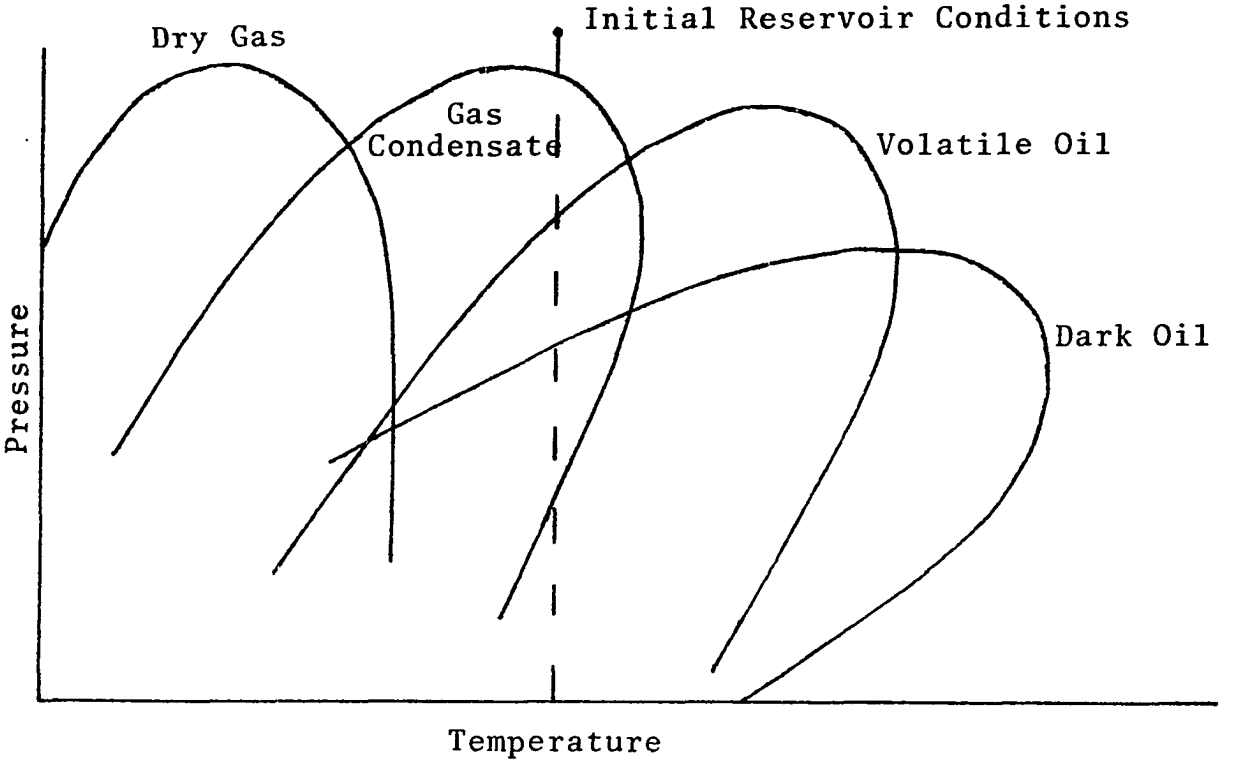



Figure 3

Types of Reservoir Systems

temperature) and are differentiated by the amount of shift of the composition, or phase diagram, which occurs during depletion. These four classifications apply only to reservoir systems which are initially in equilibrium and remain in equilibrium throughout the depletion process. This assumption of equilibrium at all times is a basic assumption used in all reservoir work. Typical compositions of each fluid type are:

| | Methane (mol %) | Intermediates (mol %) | Heptanes+ (mol %) |
|---|---|---|---|
| Dark Oil | 30 | 35 | 35 |
| Volatile Oil | 55 | 30 | 15 |
| Gas Condensate | 70 | 22 | 8 |
| Dry Gas | 90 | 9 | 1 |

Because of the serious shift of the volatile oil and retrograde gas-condensate phase diagrams during depletion, the traditional black oil simulator is not sufficiently accurate to provide good performance prediction. In these cases, and others where changes in the composition of the reservoir fluid complicate performance calculations, the compositional model is the best approach to prediction.

# CHAPTER III

# FLUID PROPERTIES

The system fluid properties are of vital importance in a reservoir simulation model and it is essential that these properties be considered not only as functions of the system temperature and pressure, but also as functions of the system composition. Many published techniques are available for predicting volumes and viscosities of hydrocarbon mixtures based on phase composition data. From these, the most widely used methods have been selected in order to ensure that the method chosen is valid over a broad range of applications. Correlations are taken directly from the literature without modification. The most direct, non-iterative techniques are employed in all cases.

## Flash Calculation

One of the most important components of a compositional reservoir simulator is the flash calculation. This is the method by which one can compute the compositions and relative amounts of the vapor and liquid phases of a given hydrocarbon mixture if one knows the overall composition of the mixture and the required equilibrium constants.

This procedure involves a simple trial-and-error material balance on a unit molal mass of the hydrocarbon mixture.

The equilibrium constant is defined as

$$K_i = \frac{y_i}{x_i} \tag{3-1}$$

where $x_i$ and $y_i$ are the mole fraction of each component in the liquid and vapor phases, respectively. The definition of mole fraction provides two more equations.

$$\sum x_i = 1.0 \tag{3-2}$$

$$\sum y_i = 1.0 \tag{3-3}$$

We also know, because there is no chemical reaction, that a molal material balance on our system must be satisfied such that the sum of the number of moles in the liquid and vapor phases must equal the total number of moles in the system, which we will call unity for convenience.

$$L + V = 1.0 \tag{3-4}$$

We can also express the overall mole fraction of a particular component in terms of the mole fractions of this component in each phase and the number of moles in each phase.

$$z_i = x_i L + y_i V \tag{3-5}$$

We can then use the definition of the equilibrium constant to eliminate the vapor mole fraction term or the liquid mole fraction term.

$$z_i = x_i(L + K_i V) \tag{3-6}$$

$$z_i = y_i(L/K_i + V) \tag{3-7}$$

Solving for each phase mole fraction and realizing that the mole fractions in each phase must sum to unity and thus the difference of the two sums must be zero, the above equations become

$$\sum_{i=1}^{n} \frac{z_i}{(L + K_i V)} - \sum_{i=1}^{n} \frac{z_i K_i}{(L + K_i V)} = 0.0 \qquad (3\text{-}8)$$

Combining the terms and using equation (3-4) to eliminate the term for the number of moles of vapor, we obtain the final form of the flash equation.

$$\sum_{i=1}^{n} \frac{z_i (1 - K_i)}{L(1 - K_i) + K_i} = 0.0 = f(L) \qquad (3\text{-}9)$$

The only unknown in this equation is the number of moles per total moles of the mixture that are in the liquid phase. The Newton-Raphson iteration technique is employed to determine the solution for the liquid split. This method is described by

$$L_{k+1} = L_k - \frac{f(L_k)}{f'(L_k)} \qquad (3\text{-}10)$$

where k is the iteration counter and L is the mole fraction in the liquid phase. The solution for the liquid phase mole fraction is that value of L which causes f(L) (equation 3-9) to equal zero.

Fortunately, the derivative of the function, which is required for the Newton-Raphson technique, is easily obtained as a direct analytical expression.

$$f'(L) = - \sum_{i=1}^{n} \frac{z_i(1 - K_i)^2}{[L(1 - K_i) + K_i]^2} \qquad (3\text{-}11)$$

Studying the above equations reveals some information which will eliminate several worries during the solution of the problem. Obvious difficulties are encountered if the denominator of any of the terms of the function or its derivative become zero, making the value of the function undefined. Thus a singularity exists for each component when

$$L(1 - K_i) + K_i = 0.0 \qquad (3\text{-}12)$$

The value of L which produces this singularity is

$$L = - \frac{K_i}{(1 - K_i)} \qquad (3\text{-}13)$$

Because the value of the equilibrium constant, $K_i$, is always, by definition, larger than zero, the values of $L_k$ at the singularity points are

$$\text{for } 0 < K_i \leq 1, \quad L_k < 0.0$$

$$\text{for } 1 < K_i, \quad L_k > 1.0$$

In the interval of physical significance ($0.0 \leq L_k \leq 1.0$), the function is well-behaved.

Noting that, by virtue of the squared terms and the non-negative mole fraction variable, the derivative function is always negative, the Newton-Raphson equation becomes

$$L_{k+1} = L_k + \frac{\sum_{i=1}^{n} \frac{z_i(1 - K_i)}{[L_k(1 - K_i) + K_i]}}{\sum_{i=1}^{n} \frac{z_i(1 - K_i)^2}{[L_k(1 - K_i) + K_i]^2}} \qquad (3\text{-}14)$$

which is solved iteratively over the interval of $0.0 \leq L_k \leq 1.0$.

To determine whether a flash calculation is required, the function is evaluated at $L_k = 0.0$ and $L_k = 1.0$.

$$f(0) = \sum_{i=1}^{n} \frac{z_i(1 - K_i)}{K_i} \qquad (3\text{-}15)$$

$$f(1) = \sum_{i=1}^{n} z_i(1 - K_i) \qquad (3\text{-}16)$$

By examining the values of these two functions and realizing that the function is always decreasing over the range of significance (i.e., the derivative is always negative), we can determine whether the system is two-phase, entirely liquid, or entirely gas. When f(0) is less than zero, the system is all gas. When f(1) is greater than zero, the system is all liquid. Otherwise, the system is two-phase and the flash calculation must be performed.

A large number of flash calculations are required during a simulator run. Thus, the faster the rate of convergence, the more efficient the simulator will be. For this reason, several different methods of obtaining an accurate initial estimate for the mole fraction of the liquid phase are tested, with the aim of obtaining a method which yields a rapid rate of convergence. Three methods of determining an initial guess for $L_o$ are tested: $L_o = 0.5$; linear interpolation between f(0) and f(1); and Lagrangian interpolation with f(0), f(0.5) and f(1). As can be seen in Table 1, the

Table 1

Rate of Convergence for Three Methods of Obtaining Initial Guess for Flash Calculation

| Liquid Split Range | Samples In Range | Iterations Required for Solution* | | | Best Method |
|---|---|---|---|---|---|
| | | 1 | 2 | 3 | |
| 0.00 - 0.05 | 5 | 31 | 29 | 28 | 3 |
| 0.05 - 0.10 | 10 | 109 | 95 | 104 | 2 |
| 0.10 - 0.15 | 13 | 140 | 143 | 79 | 3 |
| 0.15 - 0.20 | 16 | 137 | 116 | 76 | 3 |
| 0.20 - 0.25 | 10 | 58 | 54 | 38 | 3 |
| 0.25 - 0.30 | 7 | 33 | 31 | 27 | 3 |
| 0.30 - 0.35 | 4 | 17 | 16 | 13 | 3 |
| 0.35 - 0.40 | 6 | 27 | 27 | 19 | 3 |
| 0.40 - 0.45 | 6 | 20 | 28 | 16 | 3 |
| 0.45 - 0.50 | 5 | 12 | 21 | 11 | 3 |
| 0.50 - 0.55 | 2 | 6 | 8 | 4 | 3 |
| 0.55 - 0.60 | 2 | 9 | 11 | 6 | 3 |
| 0.60 - 0.65 | 3 | 11 | 11 | 7 | 3 |
| 0.65 - 0.70 | 3 | 12 | 11 | 7 | 3 |
| 0.70 - 0.75 | 2 | 8 | 6 | 4 | 3 |
| 0.75 - 0.80 | 4 | 16 | 12 | 10 | 3 |
| 0.80 - 0.85 | 3 | 12 | 9 | 6 | 3 |
| 0.85 - 0.90 | 2 | 8 | 6 | 4 | 3 |
| 0.90 - 0.95 | 0 | 0 | 0 | 0 | - |
| 0.95 - 1.00 | 2 | 8 | 4 | 3 | 3 |
| Totals | 106 | 669 | 638 | 462 | |

*Method 1 - Initial Guess = 0.50

Method 2 - Linear Interpolation Between f(0) + f(1)

Method 3 - Lagrangian Interpolation Between f(0), f(.5), + f(1)

initial guess using Lagrangian interpolation is found to be superior to the others.

For 106 samples, randomly scattered throughout the range possible for liquid split values, an average of 6.3 flash iterations are required to obtain a solution for each sample when the initial guess for the liquid split is 0.5 (the middle of the possible range for L). When using linear interpolation between f(0) and f(1) to obtain an initial guess for the value of L which satisfies the relation f(L) = 0.0, the average number of flash iterations required to obtain a solution is 6.0 for each sample. This is only a slight increase in the accuracy of the initial guess and is obtained at an overall increase in the amount of computations required. It is found that the overall efficiency of the flash routine is increased by using Lagrangian interpolation between f(0), f(.5) and f(1). An average of only 4.4 flash iterations are required for solution.

The determination of the initial guess using Lagrangian interpolation requires approximately the same number of calculations as a single flash iteration using the Newton-Raphson procedure. Many of the calculations required for the Lagrangian method are necessary for the test, discussed previously, to determine if the system is two-phase, resulting in the addition of very little computation in order to use the Lagrangian procedure. Thus, if the use of this fairly complex method of obtaining an initial guess increases the rate of convergence enough to eliminate only one

iteration, a net increase in efficiency is obtained.

Thus, the initial guess for $L_o$ is defined as

$$L_o = \frac{f(L) - f(.5)}{f(0) - f(.5)} * \frac{f(L) - f(1)}{f(0) - f(1)} * 0 + \frac{f(L) - f(0)}{f(.5) - f(0)} * \frac{f(L) - f(1)}{f(.5) - f(1)} * .5 + \frac{f(L) - f(.5)}{f(1) - f(.5)} * \frac{f(L) - f(0)}{f(1) - f(0)} * 1.0 \qquad (3\text{-}17)$$

which, realizing that at the value of L which is the desired solution, f(L) is zero, the above equation becomes

$$L_o = \frac{f(0)}{f(.5) - f(0)} * \frac{f(1)}{f(.5) - f(1)} * .5 + \frac{f(.5)}{f(1) - f(.5)} * \frac{f(0)}{f(1) - f(0)} * 1.0 \qquad (3\text{-}18)$$

This method is only used during the first flash calculation or for cells which later produce L values within 0.05 units of either boundary of the permissible range. Because only a slight change in the L value is expected for non-producing cells in the reservoir simulator, on each successive use of the flash calculation the most recent L value for that cell will be used as an initial guess.

If the Newton-Raphson technique produces a value of $L_k$ outside the permissible range, the value of $L_k$ is reset to zero or one, whichever is nearer the errant value of $L_k$.

Equilibrium K-Value Determination

The flash calculation procedure outlined previously requires values for the equilibrium constants of each

component as a function of system temperature, pressure, and convergence pressure (which is a measure of the fluid composition). The most commonly used correlations for equilibrium constants are the Natural Gas Processors Suppliers Association (NGPSA) K-value charts. These log-log charts, based on experimental data tabulated in 1947 by G. G. Brown at the University of Michigan, are available for fourteen hydrocarbon components plus nitrogen and hydrogen sulfide and at seven convergence pressures ranging from 800 psia to 10,000 psia.

The NGPSA charts and the theory of corresponding states are used as the basis of an algorithm to estimate the equilibrium ratios required in this investigation. The Canfield [2] equation is used as a basis for the study. It is of the form:

$$K = \frac{1}{P_R} \exp\left[\left\{10.7016 - \ln P_{conv} - \frac{4.4316}{T_R} - \frac{.1968}{T_R^3}\right\} \times (6.3816 - 29.0020\, Z_c + 35.3443\, Z_c^2)(1. - P_R)^{.33/T_R}\right] \quad (3\text{-}19)$$

Before development of the correlation, it would be appropriate to discuss constraints placed on K-value behavior by a viable reservoir simulation model. Fussell and Yarborough [3] have pointed out that, primarily because the derivative of the K-values with respect to both pressure and composition (expressed as convergence pressure) is used implicitly in the numerical solution to the model's basic

diffusivity equations, the simulator's results are very sensitive to the smoothness of the equilibrium ratio relationships. A further constraint, of course, is the fact that by definition the equilibrium ratio must equal 1.0 when the system pressure is equal to the convergence pressure.

Smith and Van Ness [4] indicate that for an ideal liquid solution and an ideal gas phase, the equilibrium ratio is simply and exactly the ratio of the component vapor pressure to the total system pressure. Hougen, Watson, and Ragatz [5] give an equation for the reduced component vapor pressure,

$$\log_{10} P_{RS} = \frac{-A(1 - T_R)}{T_R} - 10^{-8.68(T_R - B)^2} \qquad (3\text{-}20)$$

where

$$A = 16.25 - 73.85\ Z_c + 90.00\ Z_c^2 \qquad (3\text{-}21)$$

$$B = 1.80 - 6.20\ Z_c \qquad (3\text{-}22)$$

Using this reduced component vapor pressure in the equilibrium K-value definition, we obtain

$$K_i = \frac{P_{RS}}{P_R} \qquad (3\text{-}23)$$

The easiest correction of this equation to allow for real systems is to multiply the pressure ratio by a correction factor. Several such correction factors have been proposed in the literature, such as that set forth by Canfield [2] (who ignores the reduced component vapor pressure):

$$C = \exp\left[\left(10.7016 - \ln P_{conv} - \frac{4.4316}{T_R} - \frac{.1968}{T_R^3}\right) \times (6.3816 - 29.0020\, Z_c + 35.3443\, Z_c^2)(1 - P_R)^{.33/T_R}\right] \quad (3\text{-}24)$$

or that suggested by Hougen, Watson, and Ragatz [5]:

$$C = \exp\left[\frac{Z_c}{T_R}(.25 + .132\, T_R)(P_R - P_{RS})\right] \quad (3\text{-}25)$$

Because the forms of the two equations are similar and the Hougen, Watson, and Ragatz [5] equation is consistent in terms of the use of a reduced component vapor pressure, it is chosen as a new basis for an improved K-value correlation.

The obvious error of the Hougen, et al. [5] equation is its failure to converge to a value of 1.0 when the system pressure is equal to the convergence pressure of the system. This is corrected by including a term so that the improved equation becomes

$$K_i = \left\{\frac{P_{RS}}{P_R} \exp\left[\frac{Z_c}{T_R}(.25 + .132\, T_R)(P_R - P_{RS})\right]\right\}^{\alpha} \quad (3\text{-}26)$$

where

$$\alpha = (1 - P/P_{conv})^{\beta} \quad (3\text{-}27)$$

It is found that the value of $\beta$ is a function of the reciprocal of the reduced temperature, which agrees well with the similar term in the corresponding states K-value algorithm developed by Canfield. Optimization procedures, undertaken to determine the value of $\beta$ which yields the best

results, indicate that β is also a function of the convergence pressure. The modified equation, after optimization, becomes

$$K_i = \left\{\frac{P_{RS}}{P_R} \exp\left[\frac{Z_c}{T_R}(.25 + .132\ T_R)(P_R - P_{RS})\right]\right\}^{\alpha} \quad (3\text{-}28)$$

where

$$\alpha = (1 - P/P_{conv})^{\beta} \quad (3\text{-}29)$$

$$\beta = \frac{1}{T_R}[(1.4 * 10^{-4})P_{conv} + .08] \quad (3\text{-}30)$$

This equation is developed using 360 data points, five pure components (methane, ethane, propane, n-butane, n-pentane), three convergence pressures (1000, 2000, and 5000 psia), six pressures, and four temperatures for each component, and produces an overall average absolute error of 19.3% compared with 39.6% for the Hougen, et al. [5] equation and 37.5% for the Canfield [2] equation. Further attempts to improve this algorithm produce a highly complicated equation with only a slight improvement in accuracy.

This correlation, although superior to the Canfield [2] or Hougen, et al. [5] corresponding states algorithms, fails to provide the accuracy or smoothness which Fussell and Yarborough [3] point out is necessary for use in a compositional simulator. Several factors involved in the development of this correlation may be at fault.

The first and most obvious is the possibility that the theory of corresponding states is not valid for use in predicting equilibrium K-values. More likely, the

corresponding states correlations used as a basis in this investigation contain weaknesses which have not been corrected.

Another potential source of inaccuracies is the use of Natural Gas Processors Suppliers Association equilibrium K-value charts rather than real data as a source of values upon which this correlation was based. The smoothing, interpolation, and extrapolation involved in the NGPSA charts brings some of the accuracies into question when applied to real systems. K-values for pure component systems, in addition, cannot be accurately used to predict the vapor-liquid split of complex mixtures.

Further work on improving the K-value procedure will undoubtedly result in an improvement of the compositional simulation package.

## Convergence Pressure

The equilibrium K-value algorithm discussed requires a system convergence pressure value. Two promising techniques for evaluating the convergence pressure are available in the literature. The approach used by Simon and Yarborough [6] is highly empirical but is based on data from 14 reservoir systems. Etter and Kay [7] approach the problem by attempting to combine the pseudo-critical pressure contribution of each component and what the authors call an excess critical pressure term for each component.

MacDonald [8] investigated both methods of obtaining the system convergence pressure and compared the accuracy

of each. He indicates that for natural gases, multi-component mixtures with no $C_{7+}$, and ternary mixtures, the Etter-Kay [7] procedure produced results with an average percent error of 6.4, based on 200 examples, compared with 59.7 percent based on 129 samples obtained using the Simon-Yarborough [6] procedure. For reservoir oil systems containing $C_{7+}$, the Simon-Yarborough [6] procedure is only slightly more accurate (10.1 percent error based on 70 mixtures compared with 11.6 percent based on 75 samples). Because the Etter-Kay [7] method is at least as accurate as the Simon-Yarborough [6] procedure and, in addition, takes the system composition more heavily into account, this procedure was selected to determine the system convergence pressure to be used in the equilibrium K-value correlation.

The Etter-Kay [7] method is very straightforward and defines the convergence pressure, sometimes called the mixture critical pressure, to be

$$P_c = \sum_{i=1}^{n} Z_i P_{c_i} + \sum_{i=1}^{n-1} \Phi_i \qquad (3\text{-}31)$$

where $\Phi_i$ is the excess critical pressure due to each component and the remainder of the variables are as defined in Appendix A. The first summation is the system pseudo-critical pressure, while the second summation is the system excess critical pressure and includes all but the heaviest component. The excess critical pressure for each component is a function of the weight fraction of each component

$$w_i = \frac{M_i Z_i}{\sum_{j=1}^{n} Z_j M_j} \tag{3-32}$$

and the average molecular weight of the mixture

$$M_{av} = \sum_{j=1}^{n} w_j M_j \tag{3-33}$$

If methane is present in the system under study, its excess critical pressure is determined as a special case.

$$\Phi_1 = a_1 w_1^{b_1} (M_{av} - M_1) \tag{3-34}$$

The remainder of the component excess critical pressures are determined from the general formula

$$\Phi_j = a_j \left[\frac{Z_j}{1 - \sum_{i=1}^{j-1} Z_i}\right]^{b_j} \left[\frac{M_{av} - \sum_{i=1}^{j-1} M_i w_i}{1 - \sum_{i=1}^{j-1} w_i} - M_j\right] \left(1 - \sum_{i=1}^{j-1} Z_i\right) \tag{3-35}$$

where the constants $a_j$ and $b_j$ are defined by Etter and Kay [7] and presented in Table 2. As indicated in the table,

Table 2

Constants for Etter-Kay Convergence Pressure Technique

| i | Components | $a_j$ | $b_j$ |
|---|---|---|---|
| 1 | Methane + Nitrogen | 137. | 1.073 |
| 2 | Ethane + Carbon Dioxide | 28.6 | 1.595 |
| 3 | Propane + Hydrogen Sulfide | 13.6 | 1.225 |
| 4 | n-Butane + i-Butane | 9.24 | 1.284 |
| i>4 | | $9400\ M_i^{-1.71}$ | $2.000\ M_i^{-0.115}$ |

non-hydrocarbon constituents are included with hydrocarbons of similar molecular weights. Isomers of butane are considered to be the same component, while components containing more than four carbon atoms per molecule have constants defined as functions of the molecular weight of each.

The correlation developed by Etter and Kay [7] is limited to applications to normal paraffin mixtures of low molecular weight. MacDonald [8] has shown that extension of this procedure to heavy hydrocarbon mixtures of high molecular weight is justified if a critical pressure of 6000 psia is used for heptane-plus fractions that may be present.

Liquid Density

Yen and Woods [9] and Guerrero [10] have developed correlations for the liquid volume of a hydrocarbon mixture. The former technique is valid only for pseudo-reduced temperatures less than 1.0, a much smaller range of values than desired, while Guerrero [10] states that his procedure is not valid for reservoir conditions. Kennedy, Bowman, Crownover, and Miesch [11] have developed a very complex correlation which involves several sets of constants and a great deal of calculation. Avasthi and Kennedy [12] present improvements in the Alani and Kennedy [13] method, yet these improvements have been made at the cost of considerable increase in the complexity of the equations used and do not appear to have a significant effect on the accuracy.

The Alani-Kennedy [13] method of determining the liquid volume is selected as the best combination of high

accuracy and ease of computation. This method is based on van der Waal's equation of state which is manipulated to obtain a cubic equation, with constants defined by Alani and Kennedy [13], which can easily be solved directly using procedures available in standard math tables [14]. The authors define mixing rules for $a_m$ and $b_m$

$$a_m = \sum_{i=1}^{n} x_i a_i \tag{3-36}$$

$$b_m = \sum_{i=1}^{n} x_i b_i \tag{3-37}$$

where $a_i$ and $b_i$ are functions of temperature for pure components

$$a_i = K_i \exp[n_i/T] \tag{3-38}$$

$$b_i = m_i T + C_i \tag{3-39}$$

The values of K, n, m, and C are the constants which are defined by Alani and Kennedy [13]. If the mixture contains a heptanes-plus pseudo-component, the authors define $a_i$ and $b_i$ as

$$\ln a_{C7+} = .0038405985\, M_{C7+} - .00095638281 \frac{M_{C7+}}{\rho_{C7+}} + \frac{261.80818}{T} + 7.3104464 * 10^{-6} M_{C7+}^2 + 10.753517 \tag{3-40}$$

$$b_{C7+} = .03499274\, M_{C7+} - 7.2725403\, \rho_{C7+} + .0002232395\, T - .016322572 \frac{M_{C7+}}{\rho_{C7+}} + 6.2256545 \tag{3-41}$$

where $M_{C7+}$ is the molecular weight of the heptanes-plus

fraction, $\rho_{C7+}$ is the density in grams per cubic centimeter of the heptanes-plus fraction at standard conditions (60°F and 14.7 psia), and T is the system temperature in °R. These values are incorporated in van der Waal's equation of state in the form

$$V^3 - (\frac{RT}{P} + B_m)V^2 + \frac{A_m}{P} V - A_m B_m = 0 \qquad (3\text{-}42)$$

The liquid volume is defined as the smallest root of this cubic equation.

The authors studied 647 experimental measurements made on 47 reservoir fluid samples in the range of 72°F to 250°F and bubble point to 5000 psig. They found an average deviation of 1.6 percent with a maximum deviation of 4.9 percent.

Constants for three non-hydrocarbon components are obtained by Reamer, Sage, and Lacey [15], Van Itterbeek and Verbeke [16], and Newitt, Pai, Kuloor, and Huggil [17]. The algorithm used includes these parameters plus those developed for hydrocarbons by Alani and Kennedy [13] to ensure its applicability to most typical reservoir fluids encountered. The constants available in the literature are presented in Table 3. It should be pointed out that the constants for methane are valid for the temperature range of 70°F to 300°F, while those for ethane are valid for 100°F to 249°F. Other constants are available for both to 460°F. Because typical reservoirs are less than 250°F, it is not felt to be necessary to include the high temperature constants in the

Table 3

Coefficients Used in the Alani-Kennedy Procedure

| Component | K | n | m x $10^4$ | C |
|---|---|---|---|---|
| Carbon Dioxide | 8,166. | 126. | 1.818 | 0.3872 |
| Nitrogen | 4,300. | 2.293 | 4.49 | 0.3853 |
| Hydrogen Sulfide | 13,200. | 0. | 17.9 | 0.3945 |
| Methane | 9,160.6413 | 61.893223 | 3.3162472 | 0.50874303 |
| Ethane | 46,709.573 | -404.48844 | 5.1520981 | 0.52239654 |
| Propane | 20,247.757 | 190.24420 | 2.1586448 | 0.90832519 |
| i-Butane | 32,204.420 | 131.63171 | 3.3862284 | 1.1013834 |
| n-Butane | 33,016.212 | 146.15445 | 2.902157 | 1.1168144 |
| n-Pentane | 37,046.234 | 299.62630 | 2.1954785 | 1.4364289 |
| n-Hexane | 52,093.006 | 254.56097 | 3.6961858 | 1.5929406 |
| n-Heptane | 82,295.457 | 64.380112 | 5.2577968 | 1.7299902 |
| n-Octane | 89,185.432 | 149.39026 | 5.9897530 | 1.9310993 |
| n-Nonane | 124,062.650 | 37.917238 | 6.7299934 | 2.1519973 |
| n-Decane | 146,643.830 | 26.524103 | 7.8561789 | 2.3329874 |

package. Constants defined for the components other than methane and ethane are not limited by the authors to given temperature ranges.

When components are grouped, a new correlation for the liquid volume must be used, due to the fact that the Alani-Kennedy [13] method can be applied only to pure components. The literature indicates that a reliable, although somewat complicated, method is the Lee-Edmister modification of the Redlich-Kwong equation of state [18]. This method presents the volume as a function of the composition, critical pressures, critical temperatures, and acentric factors. The cubic equation generated is solved directly, as in the Alani-Kennedy [13] method.

The authors define constants a, b, and c in the following manner.

$$a_m = a_1 - a_2 T + a_3 T^{-1} + a_4 T^{-5} \tag{3-43}$$

$$a_j = \left(\sum y_i a_{ji}^{1/2}\right)^2 \tag{3-44}$$

$$a_{1i} = \frac{R^2 T_{ci}^2}{P_{ci}} (.25913 - .031314\ w_i) \tag{3-45}$$

$$a_{2i} = \frac{R^2 T_{ci}}{P_{ci}} (.0249 + .15369\ w_i) \tag{3-46}$$

$$a_{3i} = \frac{R^2 T_{ci}^3}{P_{ci}} (.2015 + .21642\ w_i) \tag{3-47}$$

$$a_{4i} = \frac{R^2 T_{ci}^7}{P_{ci}} (.042\ w_i) \tag{3-48}$$

$$b_m = \sum y_i \frac{RT_{ci}}{P_{ci}} (.0982) \tag{3-49}$$

$$c_m = c_1 T^{-1/2} + c_2 T^{-2} \tag{3-50}$$

$$c_j = \left(\sum y_i \, c_{ji}^{1/3}\right)^3 \tag{3-51}$$

$$c_{1i} = \frac{R^3 T_{ci}^{3.5}}{P_{ci}^2} (.059904)(1 - w_i) \tag{3-52}$$

$$c_{2i} = \frac{R^3 T_{ci}^{5}}{P_{ci}^2} (.018126 + .091944 \, w_i) \tag{3-53}$$

where variables are as defined in the Table of Nomenclature in Appendix A. These values are substituted into the Redlich-Kwong equation of state, which is rearranged to the following form

$$V^3 - \frac{RT}{P} V^2 + \left(\frac{a_m}{P} - b_m^2 - \frac{RTb_m}{P}\right)V + \frac{a_m b_m}{P} - \frac{c_m}{P} = 0 \tag{3-54}$$

This cubic equation is solved for the smallest root using the procedure presented in the standard math tables [14]. Nolen [19] found that this procedure is generally unreliable for "a priori" predictions of liquid volumes, but accurate duplication could be achieved by adjusting the critical properties and acentric factors of the heavy hydrocarbon fractions.

The Lee-Edmister [18] method is only used when several components have been grouped into one pseudocomponent. The faster Alani-Kennedy [13] method is used in those

situations where the system consists only of pure components and a heptanes-plus fraction.

The system liquid density is determined by dividing the liquid volume, obtained from one of the above correlations, into the average molecular weight of the liquid system.

Vapor Density

Many methods are available for determining the volume of a hydrocarbon vapor, although most are applicable to pure, ideal components only. Several procedures have been published which are specifically designed for complex hydrocarbon gas mixtures which are encountered in a typical petroleum reservoir. The technique developed by Standing and Katz [20], although relatively accurate, is based on a graphical approach and is not readily adaptable to systems with composition other than those cited by the authors. Kay [21] presents another graphical approach to the determination of the vapor volume. This method is not easily modified for use in a computer model.

The method selected to determine the system vapor volume was that set forth by Sarem [22], which is a curve fit of the Standing and Katz [20] compressibility factor chart as a function of the system reduced-pressure and temperature. The procedure involves the calculation of two dummy variables

$$x = \frac{2P_R - 15}{14.8} \tag{3-55}$$

$$y = \frac{2T_R - 4}{1.9} \tag{3-56}$$

The compressibility factor is then obtained from

$$z = \sum_{m=0}^{5} \sum_{n=0}^{5} A_{mn} P_m(x) P_n(y) \tag{3-57}$$

where $A_{mn}$ is an element from a 6 x 6 matrix of coefficients and P is a selection of six Legendre polynomials of degree zero through five, as defined by Sarem [22]. This compressibility factor is then incorporated into the real gas law to obtain the vapor volume for the given system. The vapor density is determined by dividing the vapor volume into the average molecular weight of the vapor system.

Viscosities

Several published techniques for determining fluid viscosities are discarded because they are valid for ternary mixtures only (Heric [23]), require values determined graphically (Carr, Kobayashi and Burrows [24]), or are not valid for complex mixtures (Dempsey [25], Lee, Starling, Dolan, and Ellington [26]).

The Little-Kennedy [27] correlation proved to be invalid for a chosen test case, although the authors indicate that some work had been done on complex liquid mixtures. The procedure developed by Lohrenz, Bray, and Clark [28] is selected for determining the liquid viscosity. Although the vapor viscosity technique developed by Lee, Gonzalez, and Eakin [29] seems to be rather simple, yet fairly accurate, the Lohrenz, et al. [28] method is also chosen to determine

the vapor viscosity. This is due to the fact that most of the calculations required for the liquid viscosity are also required for the vapor viscosity, greatly reducing the additional work involved. The Lohrenz [28] method is clearly set forth by Reid and Sherwood [30]. The Stiel-Thodos correlation [28,30] is used to determine the low pressure, pure gas viscosities for each component. This is a one time calculation and is done immediately after reading the components present in the system, using the following method

$$\mu_i = \frac{34.0 * 10^{-5} T_{R_i}^{.94}}{\tau_i}, \quad T_{R_i} \leq 1.5 \qquad (3\text{-}58)$$

$$\mu_i = \frac{17.78 * 10^{-5}(4.58 T_{R_i} - 1.67)^{.625}}{\tau_i}, \quad T_{R_i} > 1.5 \qquad (3\text{-}59)$$

where

$$\tau_i = \frac{5.44085 T_{c_i}^{1/6}}{M_i^{1/2} P_{c_i}^{2/3}} \qquad (3\text{-}60)$$

The low pressure gas mixture viscosity is calculated first using the Herning-Zipperer equation [28,30] which has the following form

$$\mu_m = \frac{\sum y_i M_i^{1/2} \mu_i}{\sum y_i M_i^{1/2}} \qquad (3\text{-}61)$$

This value is then modified to produce the actual viscosity of the mixture at reservoir conditions using the Jossi-

Stiel-Thodos correlation [28,30]

$$\mu_L = \mu_m + \{[(.1023 + .023364\left(\frac{\rho_L}{\rho_{CL}}\right) + .058533\left(\frac{\rho_L}{\rho_{CL}}\right)^2 - .040758\left(\frac{\rho_L}{\rho_{CL}}\right)^3 + .0093324\left(\frac{\rho_L}{\rho_{CL}}\right)^4 - .001]/\tau_L\} \quad (3\text{-}62)$$

and the Dean-Stiel correlation [28,30]

$$\mu_V = \mu_m + \{10.8 * 10^{-5}[\exp\left(1.439\ \frac{\rho_V}{\rho_{CV}}\right) - \exp\left(-1.111\ \left(\frac{\rho_V}{\rho_{CV}}\right)^{1.858}\right)]/\tau_V\} \quad (3\text{-}63)$$

where the variables used are as defined in the Table of Nomenclature in Appendix A. The Lohreng [28] method, a procedure incorporating several other viscosity correlations, has been used for determining the vapor and liquid viscosities of the hydrocarbon mixtures encountered during the reservoir simulation.

# CHAPTER IV

## FORMULATION OF SYSTEM EQUATIONS

An integral element of any simulator is the development of the equations which describe the system. This is also true for a compositional simulator. The development of the compositional reservoir equations is not complex, but does yield bulky equations with a great many terms. To enable these equations to be presented more clearly, various short-hand notations will be introduced during the development of the system equations.

The general phase equations for a reservoir system can be written as

$$\nabla\cdot[A\rho_j k\left(\frac{k_r}{\mu}\right)_j \nabla\Phi_j] - q_{v_j} = \frac{\partial}{\partial t}(\phi\rho_j S_j)V_k \qquad (4\text{-}1)$$

where j is a phase identifier and represents oil, gas, or water. Other variables are described in Appendix A. Also available are flow potential equations, of the form,

$$\Phi_j = P_j - \rho_j gh \qquad (4\text{-}2)$$

where, again, j is a phase identifier for oil, gas, or water. The water and gas phase pressures can be described by the oil phase pressure and the capillary pressures using the

relations

$$P_w = P_l - P_{cwo} \tag{4-3}$$

$$P_g = P_l + P_{cgo} \tag{4-4}$$

This is an area where most compositional simulators available in the literature make the simplifying assumption that the capillary pressure forces are negligible. This package will include the capillary pressure equations and will not make the assumption that all phase pressures are equal.

In addition, saturation relations are available from the definition of phase saturations which says that the sum of the saturation of each phase must be unity and, from this, that the sum of the derivatives of each phase saturation must equal zero.

These relationships provide sufficient basis for the development of a single equation which describes the compositional reservoir system. This development will now be discussed more fully.

Substituting equation (4-2) into equation (4-1), breaking up the differential on the right hand side, and dividing through by $\rho_j$ gives a new general phase equation of the form

$$\nabla[Ak\lambda_j\nabla\Phi_j] - \frac{q_{vj}}{\rho_j} = \frac{V_k}{\rho_j}\left[\phi S_j \frac{\partial\rho_j}{\partial t} + \rho_j S_j \frac{\partial\phi}{\partial t} + \rho_j\phi \frac{\partial S_j}{\partial t}\right] \tag{4-5}$$

where $\lambda_j$ is the interblock mobility term.

We can use the relation

$$\frac{\partial \rho_j}{\partial t} = \frac{\partial \rho_j}{\partial P} \frac{\partial P}{\partial t} \tag{4-6}$$

to make the substitution

$$a_j = \frac{S_j}{\rho_j} \frac{\partial \rho_j}{\partial P} \tag{4-7}$$

again where j is a phase saturation identifier. Recognizing that the compressibility of a given phase is defined as

$$C = \frac{1}{\rho} \frac{d\rho}{dP} \tag{4-8}$$

we can determine the simplifying values $a_j$ for oil, gas, and water as

$$a_j = S_j C_j \tag{4-9}$$

where $C_j$ is input to the simulator as a function of pressure in a table of lookup or polynomial curve fit.

The general equation, upon summing the general phase equation for each of the three phases, and using the saturation relations, becomes

$$\nabla\cdot[A\lambda_\ell \nabla(P_\ell - \rho_\ell gh) + A\lambda_w \nabla(P_w - \rho_w gh) + A\lambda_g \nabla(P_g - \rho_g gh)$$
$$- \left[\frac{q_{v\ell}}{\rho_\ell} + \frac{q_{vw}}{\rho_w} + \frac{q_{vg}}{\rho_g}\right] - V_k (a_\ell + a_w + a_g) \frac{\partial P}{\partial t}$$
$$- V_k \frac{\partial \phi}{\partial t} = 0 \tag{4-10}$$

By using the capillary pressure relationships, the general equation is put in a form with only the oil pressure unknown. Again incorporating variables to simplify the general

equation,

$$B_1 = a + a_w + a_g - \frac{1}{\phi}\frac{\partial \phi}{\partial P_\ell} \tag{4-11}$$

$$B_2 = \frac{q_{v\ell}}{\rho_\ell} + \frac{q_{vw}}{\rho_w} + \frac{q_{vg}}{\rho_g} \tag{4-12}$$

$$\lambda_t = \lambda_\ell + \lambda_w + \lambda_g \tag{4-13}$$

the final differential equation which describes the reservoir system is

$$\nabla\cdot[A\lambda_t\nabla(P_\ell - \rho_\ell gh) + A\lambda_w\nabla(P_{cwo} + \rho_w gh) + A\lambda_g\nabla(P_{cgo} - \rho_g gh)]$$

$$- B_2 - V_k\phi B_1\frac{\partial P_\ell}{\partial t} = 0 \tag{4-14}$$

The above equation is converted to finite difference form and analyzed using any one of several finite difference techniques which are available in the simulation package. These developments will be discussed later.

# CHAPTER V

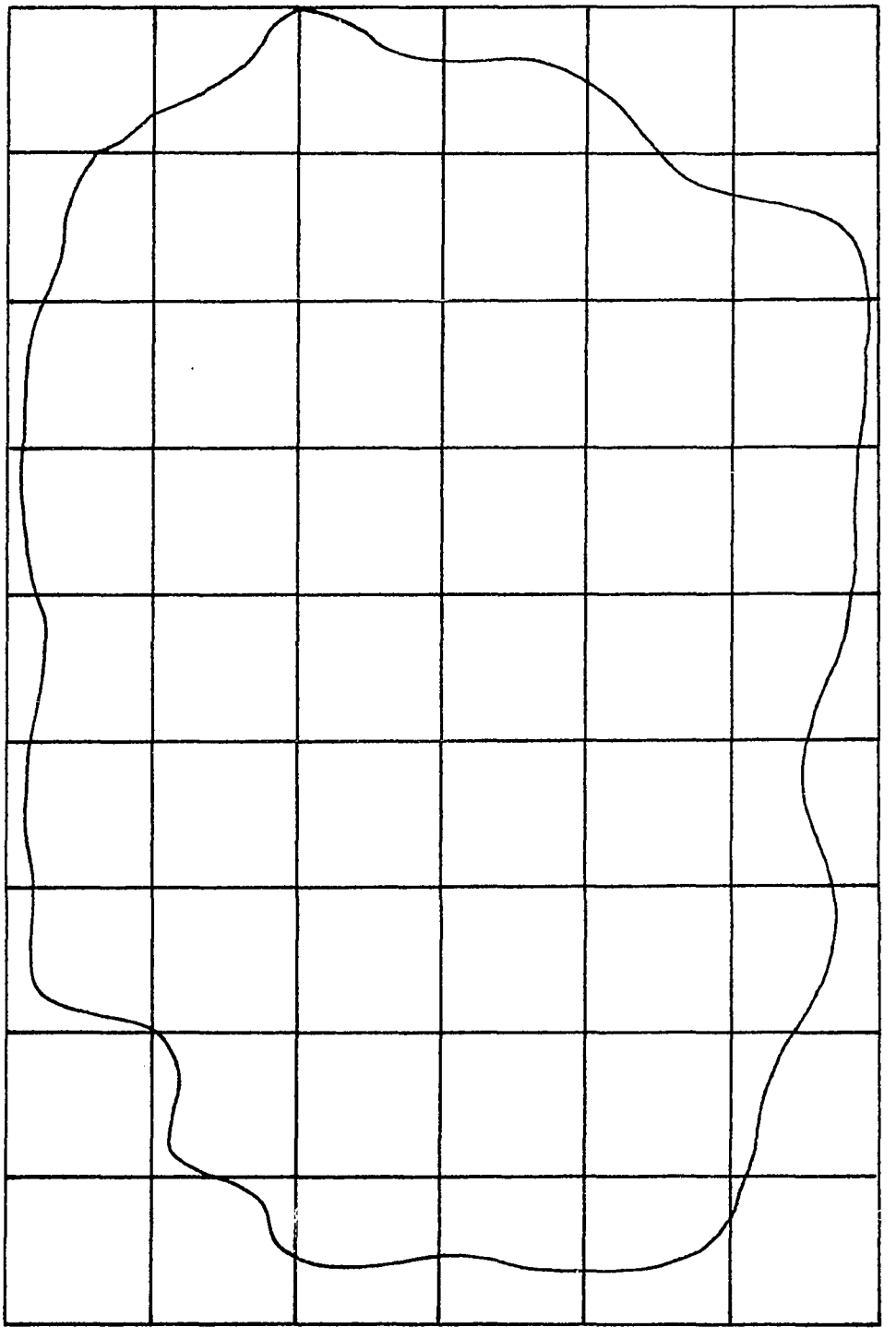
## DISCRETIZATION OF RESERVOIR SYSTEM

Except for the most trivial cases the non-linear partial differential equations which relate the oil phase pressure with time throughout the porous media of the reservoir are impossible to solve analytically. Generally, the only method of solution which can be used is a numerical formulation. While an analytical approach produces a continuous solution which exactly describes any point in the system, a numerical technique produces a discrete solution which is an approximation of the exact solution for a number of points near the same location in space or time. The transformation of the continuous differential equation to a discrete form, using finite difference techniques, will be discussed in the next chapter. The transformation of the continuous reservoir system to a discretized system is a simple procedure, but does require some explanation.

The spatial reservoir system is broken up into a number of cells or blocks by overlaying a grid system. This grid system is generally, as in this simulation package, a two-dimensional rectangular form. An oddly shaped reservoir, as shown in Figure 4, is overlayed with the desired

Figure 4

Overlay of Discrete Grid System
onto Reservoir System



grid. Because of the random shape of a typical reservoir, some of the blocks of the overlayed system may not lie within the bounds of the reservoir and these are called imaginary cells. Blocks which have at least a majority of their area within the reservoir are called real cells and are assigned values which are the average value of the area within the block.

Time must also be discretized in order to be used in numerical simulation. This is done by breaking up the total time to be studied into a number of time steps. During each of these time steps, the dependent parameter is calculated which is an approximate value of the continuous function. The time step size chosen depends on the system being considered and, as would be suspected, the smaller the time step size the more accurate the solution obtained.

This grid system is then used, along with the finite difference techniques available, to solve the given reservoir system at the discrete points in the center of each cell block.

# CHAPTER VI

## FINITE DIFFERENCE FORMULATION

Several different finite difference techniques are made available for use in the reservoir simulation package. No one technique is superior for all reservoir systems possibly encountered. For reservoirs with large areal blocks and where directional permeabilities are approximately equal, the Alternating Direction Implicit Procedure (ADIP) is the most efficient algorithm; as the degree of anisotropy increases and the areal block sizes decrease, the ADIP method breaks down and iterative methods, such as the Line Successive Over-Relaxation (LSOR) and the Iterative Alternating Direction Implicit Procedure (IADIP) techniques, are the more efficient methods of solution for the cell pressure values.

Finite difference techniques are used to solve the non-linear partial differential equation which describes the general reservoir system. This is done by making the following substitution in the general partial differential equation:

$$\frac{\partial}{\partial x}\left[A_i \frac{\partial B_i}{\partial x}\right] = \frac{2}{\Delta x_i}\left[A_{i+\frac{1}{2}}\left(\frac{B_{i+1} - B_i}{\Delta x_{i+1} + \Delta x_i}\right) - A_{i-\frac{1}{2}}\left(\frac{B_i - B_{i-1}}{\Delta x_i + \Delta x_{i-1}}\right)\right] \quad (6\text{-}1)$$

where the variables subscripted as $i + \frac{1}{2}$ or $i - \frac{1}{2}$ are the values evaluated at the boundary between the i'th and i + 1, or i'th and i - 1 cells, respectively. The discretized equation is then solved using one of the several solution techniques available.

The Alternating Direction Implicit Procedure (ADIP)

This method involves dividing each time step into two equal substeps. During the first substep, the grid system is swept in the x-direction, one row at a time, solving for the unknown pressure of each cell. During the second substep, the system is swept in the y-direction, one column at a time, solving for the unknown pressure of each cell. The pressure values obtained at the end of the second substep are the pressures for the system at the n + 1 time level. The more complex two-dimensional problem is reduced to two one-dimensional problems.

The finite difference formulation of the general system equation is manipulated to produce the tridiagonal matrix values. During the first substep, an x-direction sweep is carried out and the pressure values of the cells in the rows above (i + η) and below (i - η) are assumed known from the most recent calculation of each cell, reducing the original five unknowns to only three unknowns. The x-direction sweep equation becomes

$$A_i P_{i-1}^{n+\frac{1}{2}} + B_i P_i^{n+\frac{1}{2}} + C_i P_{i+1}^{n+\frac{1}{2}} = D_i \qquad (6\text{-}2)$$

where $A_i$, $B_i$, $C_i$, and $D_i$ are as shown in Figure 5. The

Figure 5

Coefficients of x-Direction Sweep for ADIP

$$A_i = \frac{\theta}{\Delta x_i}\left[\frac{\lambda_{xt_{i-\frac{1}{2}}}}{\Delta x_{ti-\frac{1}{2}}}\right]$$

$$B_i = \frac{\theta}{\Delta x_i}\left[\frac{-\lambda_{xt_{i+\frac{1}{2}}}}{\Delta x_{i+\frac{1}{2}}} - \frac{\lambda_{xt_{i-\frac{1}{2}}}}{\Delta x_{i-\frac{1}{2}}}\right] - \frac{V_i B_{1i}}{\Delta t}$$

$$C_i = \frac{\theta}{\Delta x_i}\left[\frac{\lambda_{xt_{i+\frac{1}{2}}}}{\Delta x_{i+\frac{1}{2}}}\right]$$

$$D_i = \frac{-\theta}{\Delta y_i}\left\{\lambda_{yt_{i+\frac{1}{2}\eta}}\frac{[(P_\ell)_{i+\eta} - (P_\ell)_i]}{\Delta y_{i+\frac{1}{2}\eta}} - \lambda_{yt_{i-\frac{1}{2}\eta}}\frac{[(P_\ell)_{i+\eta} - (P_\ell)_i]}{\Delta y_{i-\frac{1}{2}\eta}}\right.$$

$$+ \sum_j^{\ell,g,w}\left[\lambda_{yj_{i-\frac{1}{2}\eta}}\frac{[(\rho_j gh)_i - (\rho_j gh)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right.$$

$$\left.\left. - \lambda_{yj_{i+\frac{1}{2}\eta}}\frac{[(\rho_j gh)_{i+\eta} - (\rho_j gh)_i]}{\Delta y_{i+\frac{1}{2}\eta}}\right] + \lambda_{yw_{i-\frac{1}{2}\eta}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}}}\right.$$

$$- \lambda_{yw_{i+\frac{1}{2}\eta}}\frac{[(P_{cwo})_{i+\eta} - (P_{cwo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} + \lambda_{yg_{i+\frac{1}{2}\eta}}\frac{[(P_{cgo})_{i+\eta} - (P_{cgo})_i]}{\Delta y_{i+\frac{1}{2}\eta}}$$

$$+ \lambda_{yg_{i-\frac{1}{2}\eta}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} - \frac{\theta}{\Delta x_i}\sum_j^{\ell,g,w}\left[\lambda_{xj_{i-\frac{1}{2}}}\right.$$

$$\left.\times \frac{[\rho_j gh)_i - (\rho_j gh)_{i-1}]}{\Delta x_{i-\frac{1}{2}}} - \lambda_{xj_{i+\frac{1}{2}}}\frac{[(\rho_j gh)_{i+1} - (\rho_j gh)_i]}{\Delta x_{i+\frac{1}{2}}}\right]$$

$$+ \lambda_{xw_{i-\frac{1}{2}}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}} - \lambda_{xw_{i+\frac{1}{2}}}\frac{[(P_{cwo})_{i+1} - (P_{cwo})_i]}{\Delta x_{i+\frac{1}{2}}}$$

$$\left. + \lambda_{xg_{i+\frac{1}{2}}}\frac{[(P_{cgo})_{i+1} - (P_{cgo})_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xg_{i+\frac{1}{2}}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right\}$$

$$+ OT_i$$

variable $OT_i$ consists of all of the values based on the last time step and is determined from the equation shown in Figure 6.

During the second substep, a y-direction sweep is carried out and the pressure values of the cells in the columns to the left (i - 1) and right (i + 1) are assumed known from the most recent calculation of each cell. The y-direction sweep equation has three unknowns and becomes

$$A_i P_{i-\eta}^{n+1} + B_i P_i^{n+1} + C_i P_{i+\eta}^{n+1} = D_i \qquad (6\text{-}3)$$

where $A_i$, $B_i$, $C_i$, and $D_i$ are as shown in Figure 7. The x-direction sweep yields a close band tridiagonal matrix system when written for each cell in the model, while the y-direction sweep yields a wide band tridiagonal matrix system. The close band matrix is solved using the Thomas algorithm at the end of the x-direction sweep and the wide band matrix is solved using a modified form of the Thomas algorithm at the end of the y-direction sweep.

The Thomas algorithm is a standard technique for solving a tridiagonal matrix system and involves a forward sweep and a backward sweep of the matrix. The forward sweep for a close band matrix system involves the following procedure:

$$C_1 = C_1/B_1 \qquad (6\text{-}4)$$

$$D_1 = D_1/B_1 \qquad (6\text{-}5)$$

$$C_i = C_i/(B_i - A_i*C_{i-1}),\ i = 2,n \qquad (6\text{-}6)$$

$$D_i = D_i/(B_i - A_i*C_{i-1}),\ i = 2,n \qquad (6\text{-}7)$$

Figure 6

Previous Time Level Equation Used In Finite Difference Techniques

$$
\begin{aligned}
OT_i = \frac{(\theta - 1)}{\Delta x_i}\Bigg\{ & \lambda_{xt_{i+\frac{1}{2}}} \frac{[(P_\ell)_{i+1} - (P_\ell)_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xt_{i-\frac{1}{2}}} \frac{[(P_\ell)_i - (P_\ell)_{i-1}]}{\Delta x_{i-\frac{1}{2}}} \\
& + \sum_j^{\ell,g,w} \Bigg[ \lambda_{xj_{i-\frac{1}{2}}} \frac{[(\rho_j gh)_i - (\rho_j gh)_{i-1}]}{\Delta x_{i-\frac{1}{2}}} \\
& - \lambda_{xj_{i+\frac{1}{2}}} \frac{[(\rho_j gh)_{i+1} - (\rho_j gh)_i]}{\Delta x_{i+\frac{1}{2}}} + \lambda_{yj_{i-\frac{1}{2}\eta}} \frac{[(\rho_j gh)_i - (\rho_j gh)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \\
& - \lambda_{yj_{i+\frac{1}{2}\eta}} \frac{[(\rho_j gh)_{i+} - (\rho_j gh)_i]}{\Delta y_{i+\frac{1}{2}\eta}} \Bigg] + \lambda_{xw_{i-\frac{1}{2}}} \frac{[(P_{cwo})_i - (P_{cwo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}} \\
& - \lambda_{xw_{i+\frac{1}{2}}} \frac{[(P_{cwo})_{i+1} - (P_{cwo})_i]}{\Delta x_{i+\frac{1}{2}}} + \lambda_{xg_{i+\frac{1}{2}}} \frac{[(P_{cgo})_{i+1} - (P_{cgo})_i]}{\Delta x_{i+\frac{1}{2}}} \\
& - \lambda_{xg_{i-\frac{1}{2}}} \frac{[(P_{cgo})_i - (P_{cgo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}} \Bigg\} + \frac{(\theta - 1)}{\Delta y_i} \Bigg\{ \lambda_{yt_{i+\frac{1}{2}\eta}} \\
& \times \frac{[(P_\ell)_{i+\eta} - (P_\ell)_i]}{\Delta y_{i+\frac{1}{2}\eta}} - \lambda_{yt_{i-\frac{1}{2}\eta}} \frac{[(P_\ell)_i - (P_\ell)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \\
& + \sum_j^{\ell,g,w} \Bigg[ \lambda_{yj_{i-\frac{1}{2}\eta}} \frac{[(\rho_j gh)_i - (\rho_j gh)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \\
& - \lambda_{yj_{i+\frac{1}{2}\eta}} \frac{[(\rho_j gh)_{i+\eta} - (\rho_j gh)_i]}{\Delta y_{i+\frac{1}{2}}} \Bigg] + \lambda_{yw_{i-\frac{1}{2}}} \frac{[(P_{cwo})_i - (P_{cwo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \\
& - \lambda_{yw_{i+\frac{1}{2}\eta}} \frac{[(P_{cwo})_{i+\eta} - (P_{cwo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} + \lambda_{yg_{i+\frac{1}{2}\eta}} \frac{[(P_{cgo})_{i+\eta} - (P_{cgo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} \\
& - \lambda_{yg_{i-\frac{1}{2}\eta}} \frac{[(P_{cgo})_i - (P_{cgo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \Bigg\} + B_2 - \frac{V_i B_{1i}}{\Delta t} (P_\ell)_i
\end{aligned}
$$

Figure 7

Coefficients of y-Direction Sweep for ADIP

$$A_i = \frac{\theta}{\Delta y_i}\left[\frac{\lambda_{yt_{i-\frac{1}{2}\eta}}}{\Delta y_{i-\frac{1}{2}\eta}}\right]$$

$$B_i = \frac{\theta}{\Delta y_i}\left[\frac{-\lambda_{yt_{i+\frac{1}{2}\eta}}}{\Delta y_{i+\frac{1}{2}\eta}} - \frac{\lambda_{yt_{i-\frac{1}{2}\eta}}}{\Delta y_{i-\frac{1}{2}\eta}}\right] - \frac{V_i B_{li}}{\Delta t}$$

$$C_i = \frac{\theta}{\Delta y_i}\left[\frac{\lambda_{yt_{i+\frac{1}{2}\eta}}}{\Delta y_{i+\frac{1}{2}\eta}}\right]$$

$$\begin{aligned}
D_i = {} & \frac{-\theta}{\Delta x_i}\left\{\lambda_{xt_{i+\frac{1}{2}}}\frac{[(P_\ell)_{i+1} - (P_\ell)_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xt_{i-\frac{1}{2}}}\frac{[(P_\ell)_i - (P_\ell)_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right. \\
& + \sum_j^{\ell,g,w}\left[\lambda_{xj_{i-\frac{1}{2}}}\frac{[(\rho_j gh)_i - (\rho_j gh)_{i-1}]}{\Delta x_{i-\frac{1}{2}}} - \lambda_{xj_{i+\frac{1}{2}}}\frac{[(\rho_j gh)_{i+1} - (\rho_j gh)_i]}{\Delta x_{i+\frac{1}{2}}}\right] \\
& + \lambda_{xw_{i-\frac{1}{2}}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}} - \lambda_{xw_{i+\frac{1}{2}}}\frac{[(P_{cwo})_{i+1} - (P_{cwo})_i]}{\Delta x_{i+\frac{1}{2}}} \\
& \left. + \lambda_{xh_{i+\frac{1}{2}}}\frac{[(P_{cgo})_{i+1} - (P_{cgo})_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xg_{i-\frac{1}{2}}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right\} \\
& - \frac{\theta}{\Delta y_i}\left\{\sum_j^{\ell,g,w}\left[\lambda_{yj_{i-\frac{1}{2}\eta}}\frac{[(\rho_j gh)_i - (\rho_j gh)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right.\right. \\
& \left. - \lambda_{yj_{i+\frac{1}{2}\eta}}\frac{[(\rho_j gh)_{i+\eta} - (\rho_j gh)_i]}{\Delta y_{i+\frac{1}{2}\eta}}\right] + \lambda_{yw_{i-\frac{1}{2}\eta}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} \\
& - \lambda_{yw_{i+\frac{1}{2}\eta}}\frac{[(P_{cwo})_{i+\eta} - (P_{cwo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} + \lambda_{yg_{i+\frac{1}{2}}}\frac{[(P_{cgo})_{i+\eta} - (P_{cgo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} \\
& \left. - \lambda_{yg_{i-\frac{1}{2}\eta}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right\} + OT_i
\end{aligned}$$

where n is the number of cells in the system and A, B, C, and D are the arrays of elements of the tridiagonal system as shown in Figure 8. The backward sweep is

$$D_{i-1} = D_{i-1} - C_{i-1}*D_i, \quad i = n,1 \qquad (6\text{-}8)$$

The solution to the close band tridiagonal matrix system is thus the new matrix D.

The Thomas algorithm is slightly modified to handle a wide band matrix system, one in which diagonal arrays A and C are separated from diagonal array B by one or more zero diagonals. The forward sweep becomes

$$A_i = C_i/B_i , \qquad i = 1,\eta \qquad (6\text{-}9)$$

$$D_i = D_i/B_i , \qquad i = 1,\eta \qquad (6\text{-}10)$$

$$A_i = C_i/(B_i - A_i*A_{i-\eta}), \qquad i = \eta+1,n \qquad (6\text{-}11)$$

$$D_i = (D_i - A_i*D_{i-\eta})/(B_i - A_i*A_{i-\eta}), \qquad i = \eta+1,n \qquad (6\text{-}12)$$

The backward sweep is

$$D_i = D_i - A_i*D_{i-1} , \qquad i = n-\eta,1 \qquad (6\text{-}13)$$

Again, the solution to the wide band tridiagonal matrix system is the new matrix D.

## The Iterative Alternating Direction Implicit Procedure (IADIP)

This iterative method is similar to the ADI procedure discussed above but involves the addition of a term called the acceleration parameter. Beginning with the same general, two-dimensional system equation for the oil phase pressure,

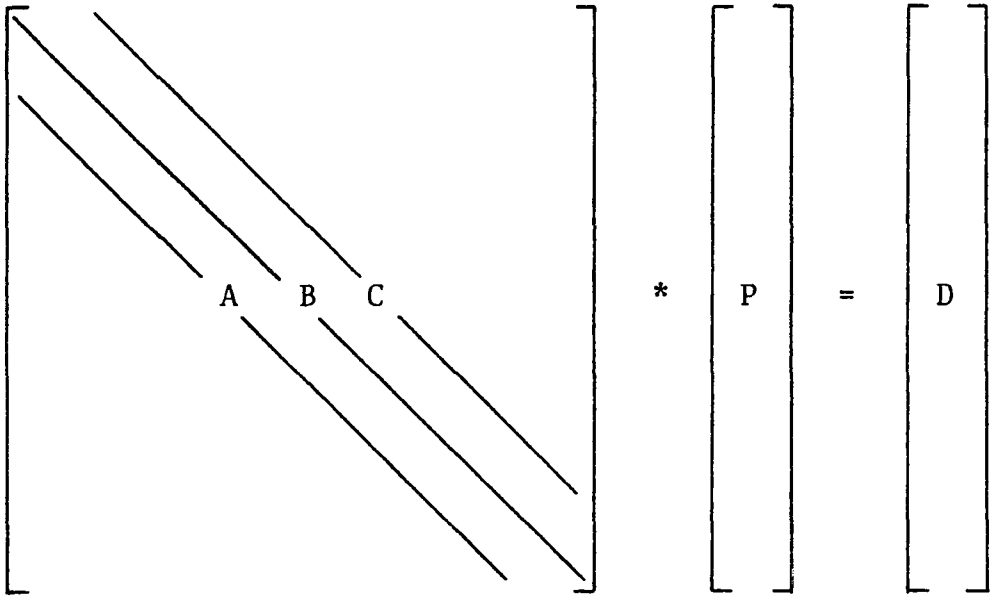



Figure 8

Close Band Tridiagonal Matrix

the same finite difference formulation is obtained except for the addition of the acceleration parameter term ($H_k$). With the addition of the acceleration parameter, the x-direction sweep equation becomes

$$A_i P_{i-1}^{k+1,\,n+\frac{1}{2}} + B_i P_i^{k+1,\,n+\frac{1}{2}} + C_i P_{i-1}^{k+1,\,n+\frac{1}{2}} = D_i \qquad (6\text{-}14)$$

where $A_i$ and $C_i$ are the same as defined for the x-direction sweep of the ADI procedure and

$$B_i = B_i(\text{x-ADIP}) - H_k \qquad (6\text{-}15)$$

$$D_i = D_i(\text{x-ADIP}) - H_k[P_\ell)_i^n] \qquad (6\text{-}16)$$

The close band form of the Thomas algorithm is again used to solve the generated tridiagonal matrix system.

The y-direction sweep equation becomes

$$A_i P_{i-\eta}^{k+1,\,n+1} + B_i P_i^{k+1,\,n+1} + C_i P_{i+\eta}^{k+1,\,n+1} = D_i \qquad (6\text{-}17)$$

where

$$B_i = B_i(\text{y-ADIP}) - H_k \qquad (6\text{-}18)$$

$$D_i = D_i(\text{y-ADIP}) - H_k[(P_\ell)_i^n] \qquad (6\text{-}19)$$

and $A_i$ and $C_i$ are as defined in the y-direction sweep of the ADI procedure. The tridiagonal matrix system is solved, as before, using the wide band form of the Thomas algorithm

The successive iterative use of a set of acceleration parameters constitutes a cycle. Equations (6-14) and (6-17) are solved iteratively, using a different acceleration

parameter for each iteration. At the end of each cycle a relative error check is made.

An optimal set of acceleration parameters is necessary to achieve the superior efficiency of this method. The determination of a set of optimum acceleration parameters is difficult analytically and is usually achieved using trial and error techniques. The upper and lower limits of the parameter can be estimated using the method of Peaceman and Rachford [31]. It has been pointed out that a geometric series of acceleration parameters produces optimum results [31]. A subroutine to estimate the acceleration parameters is available to the operator of this simulator and is discussed more fully in Chapter VII.

The acceleration parameters must be normalized to account for possible differences in the areal size of any one cell of the model with respect to the others.

$$H_k = H_k \lambda_i \tag{6-20}$$

$$\lambda_m = \left(\frac{1}{n} \sum_{i=1}^{n} K_i\right)\left(\frac{.22}{\Delta x_m \Delta y_m}\right) \tag{6-21}$$

The acceleration parameters are normalized within the IADIP computation.

Line Successive Over-Relaxation (LSOR)

This method is similar to the ADI procedure except that a solution sweep of the model occurs in one direction rather than two. The system is swept in the x-direction, row by row. Beginning with the general, two-dimensional

form of the pressure equation, the finite difference form of the equation is identical to that obtained for the ADI procedure. With only one time substep, the x-direction sweep becomes

$$A_i P_{i-1}^{n+1,\,k+1} + B_i P_i^{n+1,\,k+1} + C_i P_{i+1}^{n+1,\,k+1} = D_i \qquad (6\text{-}22)$$

where $A_i$, $B_i$, $C_i$, and $D_i$ are as given in Figure 9. The variable $OT_i$ is a combination of all values based on the previous time step and is shown in Figure 7.

At the completion of each sweep of the system, if the relative error check is not satisfied, the updated pressure estimates are over-relaxed to produce more accurate updated values.

$$P_i^{n+1,\,k+1} = P_i^{n+1,\,k} + \omega(P_i^{n+1,\,k+1} - P_i^{n+1,\,k}) \qquad (6\text{-}23)$$

An optimum relaxation parameter ($\omega$) is necessary for maximum efficiency of this method. The selection of this optimum value requires a trial and error approach. Several abbreviated simulation runs are made with various relaxation parameters. The number of iterations required for convergence is then plotted versus the relaxation parameter. A typical plot of this relationship is shown in Figure 10. This graph has the characteristic shape of a cusp, with the low point of this cusp being the optimum relaxation parameter. As a general rule, the more homogeneous the system, the closer the value of the relaxation parameter is to 1.0, while

Figure 9

Coefficients Used in the LSOR Procedure

$$A_i = \frac{\theta}{\Delta x_i}\left[\frac{\lambda_{xt_{i-\frac{1}{2}}}}{\Delta x_{i-\frac{1}{2}}}\right]$$

$$B_i = \frac{\theta}{\Delta x_i}\left[\frac{-\lambda_{xt_{i+\frac{1}{2}}}}{\Delta x_{i+\frac{1}{2}}} - \frac{\lambda_{xt_{i-\frac{1}{2}}}}{\Delta x_{i-\frac{1}{2}}}\right] - \frac{\theta}{\Delta y_i}\left[\frac{\lambda_{yt_{i+\frac{1}{2}\eta}}}{\Delta y_{i+\frac{1}{2}\eta}} + \frac{\lambda_{yt_{i-\frac{1}{2}\eta}}}{\Delta y_{i-\frac{1}{2}\eta}}\right] - \frac{V_i B_{1i}}{\Delta t}$$

$$C_i = \frac{\theta}{\Delta x_i}\left[\frac{\lambda_{xt_{i+\frac{1}{2}}}}{\Delta x_{i+\frac{1}{2}}}\right]$$

$$D_i = \frac{-\theta}{\Delta y_i}\left\{\lambda_{yt_{i+\frac{1}{2}}}\frac{(P_\ell)_{i+\eta}}{\Delta y_{i+\frac{1}{2}\eta}} + \lambda_{yt_{i-\frac{1}{2}\eta}}\frac{(P_\ell)_{i-\eta}}{\Delta y_{i-\frac{1}{2}\eta}} + \sum_j^{\ell,g,w}\left[\lambda_{yj_{i-\frac{1}{2}\eta}}\right.\right.$$

$$\times \frac{[(\rho_j gh)_i - (\rho_j gh)_{i+\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} - \lambda_{yj_{i+\frac{1}{2}\eta}}\left.\frac{[(\rho_j gh)_{i+\eta} - (\rho_j gh)_i]}{\Delta y_{i+\frac{1}{2}\eta}}\right]$$

$$+ \lambda_{yw_{i-\frac{1}{2}\eta}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}} - \lambda_{yw_{i+\frac{1}{2}\eta}}\frac{[(P_{cwo})_{i+\eta} - (P_{cwo})_i]}{\Delta y_{i+\frac{1}{2}\eta}}$$

$$\left.+ \lambda_{yg_{i+\frac{1}{2}\eta}}\frac{[(P_{cgo})_{i+\eta} - (P_{cgo})_i]}{\Delta y_{i+\frac{1}{2}\eta}} - \lambda_{yg_{i-\frac{1}{2}\eta}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right\}$$

$$- \frac{\theta}{\Delta x_i}\left\{\sum_j^{\ell,g,w}\left[\lambda_{xj_{i-\frac{1}{2}}}\frac{[(\rho_j gh)_i - (\rho_j gh)_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right.\right.$$

$$\left.- \lambda_{xj_{i+\frac{1}{2}}}\frac{[(\rho_j gh)_{i+1} - (\rho_j gh)_i]}{\Delta x_{i+\frac{1}{2}}}\right] + \lambda_{xw_{i-\frac{1}{2}}}\frac{[(P_{cwo})_i - (P_{cwo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}}$$

$$- \lambda_{xw_{i+\frac{1}{2}}}\frac{[(P_{cwo})_{i+1} - (P_{cwo})_i]}{\Delta x_{i+\frac{1}{2}}} + \lambda_{xg_{i+\frac{1}{2}}}\frac{[(P_{cgo})_{i+1} - (P_{cgo})_i]}{\Delta x_{i+\frac{1}{2}}}$$

$$\left.- \lambda_{xg_{i-\frac{1}{2}}}\frac{[(P_{cgo})_i - (P_{cgo})_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right\} + OT_i$$

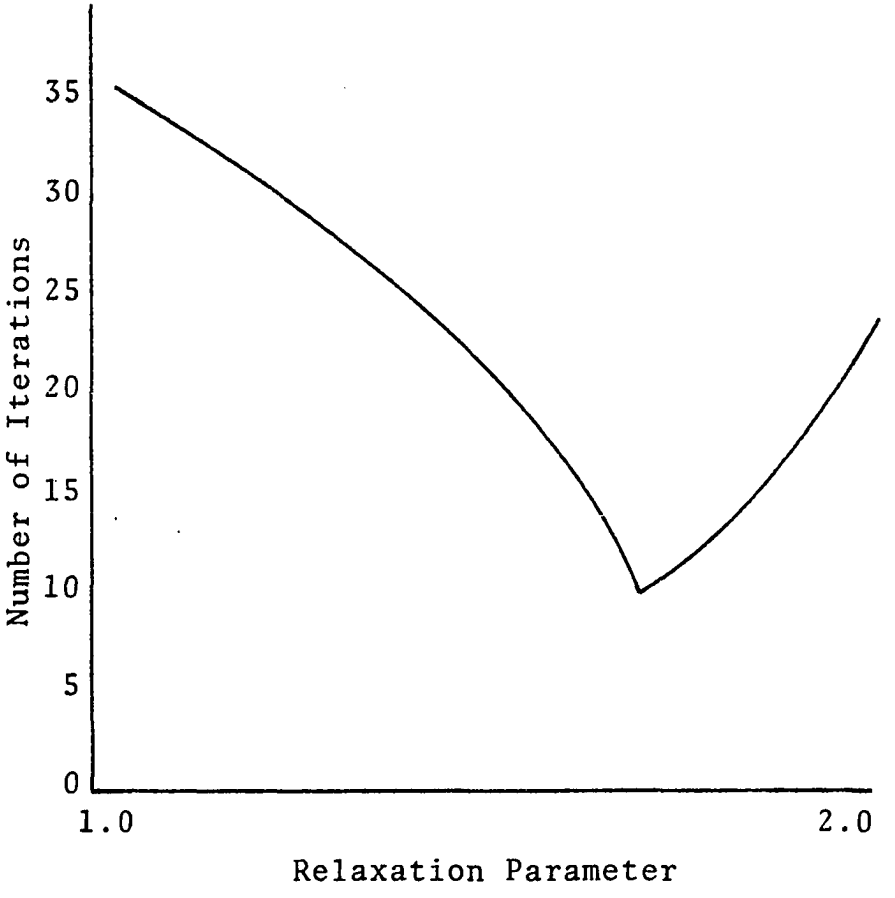



Figure 10

Typical Plot of Relaxation Parameter

the value approaches 2.0 as the degree of anisotropy of the system increases. As can be seen from Figure 10, an overestimation of the relaxation parameter is always preferred to an underestimation.

These three finite difference solution techniques, ADIP, IADIP, and LSOR, are programmed such that the user has the option of choosing the solution technique to be used to solve the oil phase pressure equation.

# CHAPTER VII

## AUXILIARY PORTIONS OF SIMULATION PACKAGE

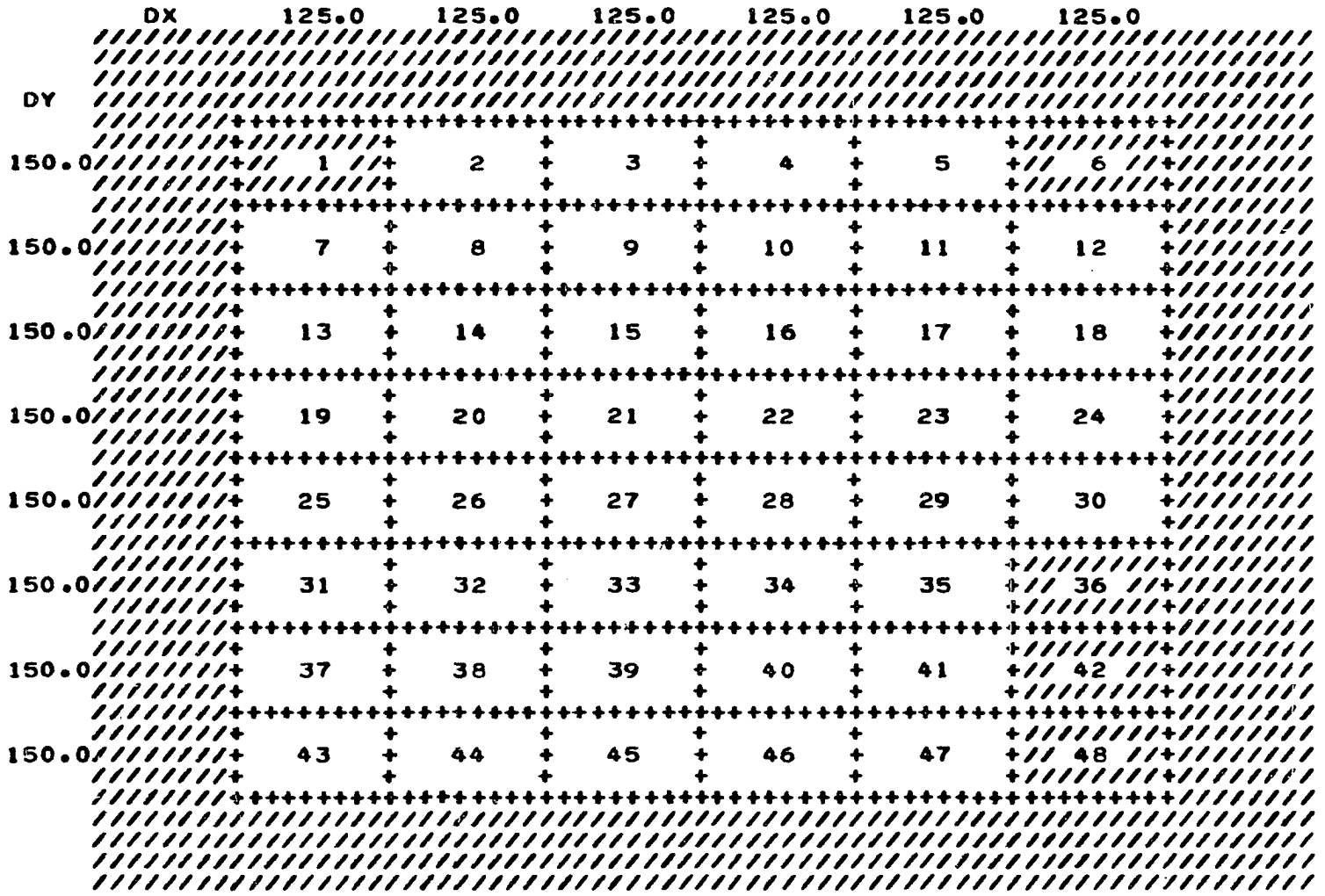
A number of auxiliary routines have been included in the simulation package, developed either out of necessity or simply for the convenience of the engineer who digitizes the data and analyzes the results. Each of these routines will now be discussed in detail. Flow charts for these routines and all others used in this simulation package are presented in Appendix C.

### WRTMOD

This routine produces a sketch, similar to that presented in Figure 11, of the system under study. When an odd shaped reservoir is discretized for analysis in a simulator, as was discussed in Chapter V, often some of the rectangular cells are not within the bounds of the system. In addition, each of the cells in the system are numbered as they are identified by the simulator. The dimensions of each cell are also shown: the x-direction dimension across the top of the sketch and the y-direction dimension down the left side.

FIGURE 11

NUMBERING SCHEME OF RESERVOIR MODEL (IMAGINARY CELLS SHADED)



This routine serves as a check for the engineer that the discretized grid has the desired shape and provides a convenient way for him to rapidly identify separate simulation runs on several different reservoir configurations.

Information required for this routine is the x and y dimensions for each cell, the number of columns in the system, the total number of cells in the system, and the cell numbers of those cells which are outside the actual boundaries of the system.

ACCP

This routine provides the option for the user to call upon the simulation package to estimate the proper acceleration parameters to be used in the Iterative Alternating Direction Implicit Procedure (IADIP). The procedure used to obtain these estimates of acceleration parameters is outlined by Crichlow [31].

Four parameters are defined as functions of the x- and y-direction interblock transmissibilities ($T_x$ and $T_y$) and the number of cells in the x- and y-direction ($N_x$ and $N_y$) in the following manner

$$M_1 = \frac{2T_x}{T_x + T_y} \frac{\pi^2}{4N_x^2} \tag{7-1}$$

$$M_2 = \frac{2T_x}{T_x + T_y} \tag{7-2}$$

$$M_3 = \frac{2T_y}{T_x + T_y} \frac{\pi^2}{4N_y^2} \tag{7-3}$$

$$M_4 = \frac{2T_y}{T_x + T_y} \tag{7-4}$$

The lower limit of the acceleration parameters is the smaller of $M_1$ and $M_3$ while the upper limit is the larger of $M_2$ and $M_4$. Birkoff, Varga, and Yound [32] point out that the acceleration parameters should be in a geometric series such that each element in the series is a constant multiplied by the previous element. The constant multiplier is determined as

$$r = \exp\left[\frac{\ln(H_m/H_1)}{M - 1}\right] \tag{7-5}$$

where $H_1$ and $H_m$ are the minimum and maximum acceleration parameters, respectively, and M is the number of acceleration parameters in the system. Each of the acceleration parameters can now be determined by multiplying the previous value by the constant.

This procedure is only a method of estimating the proper acceleration parameters to be used. The user must experiment to some extent in order to obtain an adequate lower value. The rate of convergence is very sensitive to this value and a slight difference in the lower acceleration parameter could result in divergence rather than convergence.

INPUT

This routine provides a means of reading a data matrix into the simulator, assigning a value to each cell of the system, and checking that the value assigned is within

the range specified for all of the matrix values. In addition, the entire matrix can be multiplied by a given factor. This latter feature is very important when one is attempting to obtain a history match in order to provide a realistic contour map of the rock properties for the system. The data matrix is written out to aid the user and any cell data which has been mispunched and lies outside the range specified for the system is printed out so that it can be easily corrected.

INDATA

The purpose of this routine is to read that data which is available as a table of lookup or a polynomial. If the data is in the form of a table of lookup, the independent and dependent variables are read and printed and an error message is provided if the independent variables in the table of lookup are not in ascending order. If the data is available in the form of a polynomial curve fit, the coefficients are read in order of increasing degree, beginning with the coefficient of the zero'th degree term. Flags are also associated with the data to indicate whether they are for a polynomial equation, a two point linear interpolation table of lookup, or a three point Lagrangian table of lookup and for the number of coefficients or sets of independent and dependent values which are provided. Tables of lookup or polynomial curve fits are typically used to describe relative permeabilities or formation volume factors which are required in a simulation package.

LOOKUP

This routine is designed to determine, given the value of the independent variable, the respective value of the dependent variable from a table of lookup or polynomial equation. The polynomial coefficients or table of lookup information are read into the simulation package using subroutine INDATA and subroutine LOOKUP determines from the appropriate flag which procedure is to be used to determine the dependent variable value desired. Either a two point linear interpolation or a three point Lagrangian interpolation procedure is available for use on table of lookup data. If the independent value submitted is out of the range of the table of lookup, an error message is printed and execution of the simulator is terminated at that point.

SORSNK

This routine provides a means for determining the flow rate at source or sink cells. The two basic wellbore conditions are flow at constant pressure conditions or flow at a constant rate. This simulation package is designed to utilize the explicit production procedure which requires that the oil, gas, and water flow rates be known a priori in order to set up the complete finite difference formulation. The explicit production formulation has the disadvantage that saturation oscillations or fluctuations in the flow rates can be produced if small grid blocks are used, large time steps are required, the reservoir exhibits a converging

flow pattern (i.e., fractures or other communications are present which increase flow rates abnormally), or extremely high production or injection rates are present. The implicit production procedure, which requires that the three phase flow rates be included in the implicit formulations of the finite difference equations, provides a safer method of handling the flow rates, but requires greater computer storage and longer execution of the simulation package. As long as the user is aware of the limitations imposed by the explicit procedure, only minimal, if any, oscillation problems should be encountered.

The formulation used in the simulation package requires either the flowing bottom hole pressure of the well or the phase volume produced of one phase, the oil, gas, or water phase, in that order. If the gas phase is given, for example, the oil phase flow rate is assumed to be zero and the water phase flow rate is calculated based on the mobility ratios and the given gas flow rate. If the flowing bottom hole pressure is given, the oil phase flow rate is calculated using the following equation:

$$q_o = \frac{7.07\ k_{ro} Kh(\bar{P} - P_{wf})}{\mu_o B_o \left[\ln\left(\frac{1}{r_w}\sqrt{\frac{\Delta x \Delta y}{\pi}}\right) - \frac{3}{4} + S\right]} \qquad (7\text{-}6)$$

where $\bar{P}$ is the average cell pressure and $r_w$ is the wellbore radius and is assumed to be .667 feet for all wells.

The new time level flow rates are calculated from the following set of equations:

$$q_o^{n+1} = q_o \tag{7-7}$$

$$q_g^{n+1} = \frac{k_{rg}}{\mu_g B_g} \frac{\mu_o B_o}{k_{ro}} q_o^{n+1} \tag{7-8}$$

$$q_w^{n+1} = \frac{k_{rw}}{\mu_w B_w} \frac{\mu_g B_g}{k_{rg}} q_o^{n+1} \tag{7-9}$$

BOUND

This routine examines a given cell and determines, from its position in the grid system, what values should be used in the finite difference equation for each of the four surrounding cells. This is necessary, for example, when one or more of the cell boundaries lie on the system boundary. If the given cell is not completely surrounded by other cells in the system, boundary conditions must be satisfied such that there is no flow across the system boundaries. The values used for real cells surrounding the cell of interest are the values of the surrounding cells, but, for imaginary cells surrounding the cell of interest, the value of the cell of interest is used to ensure no flow to or from any imaginary cells.

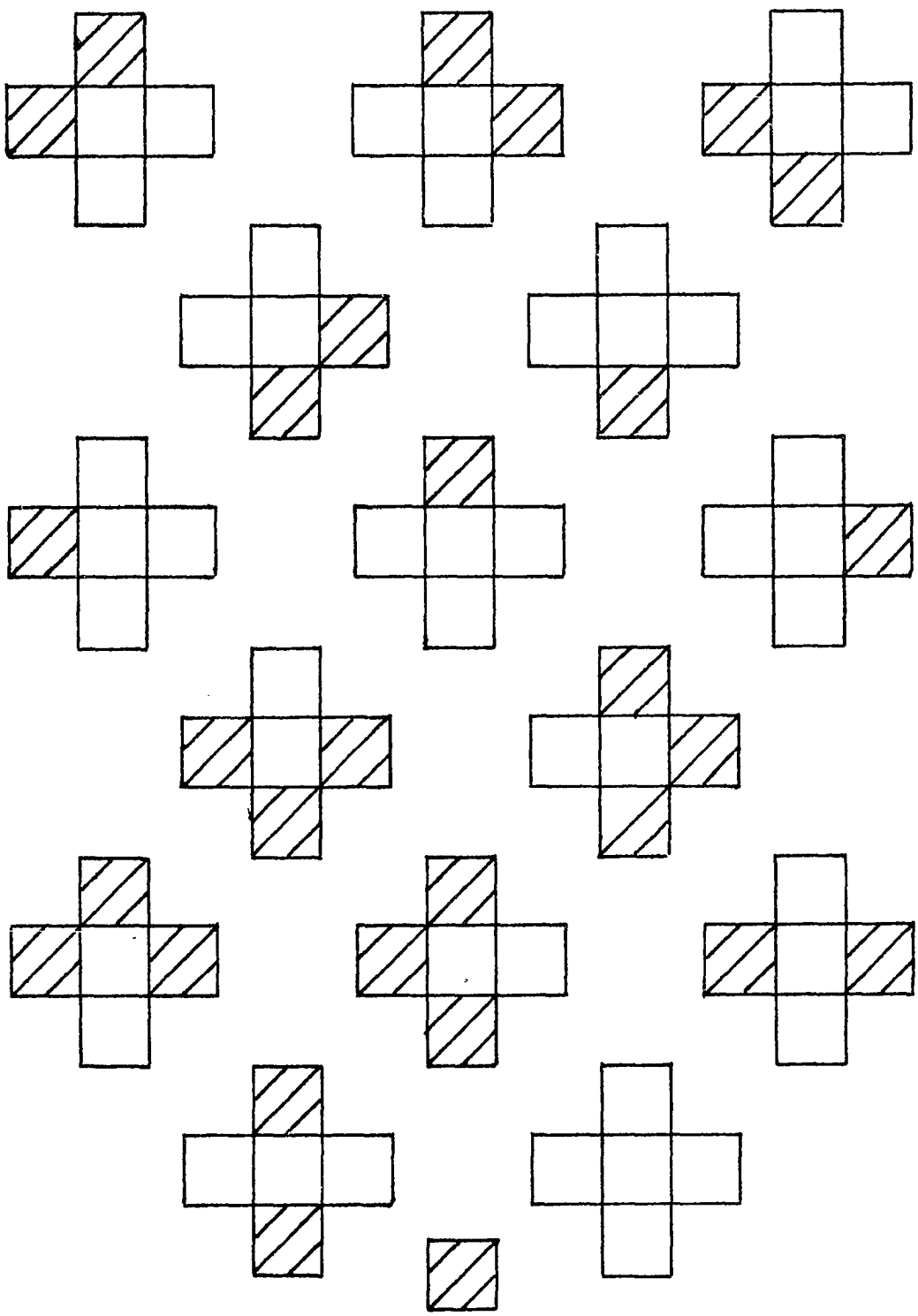
Each of the cells in the system is assigned an identifying number which indicates the configuration of it and the four surrounding cells. There are sixteen possible configurations and they are presented in Figure 12.

CUBEQN

This routine, required for use in the liquid volume correlations discussed previously, provides the smallest

Figure 12

Possible Cell Configurations



positive real root to a given cubic equation. The procedure, as outlined in the CRC Standard Mathematical Tables [14], is very simple.

The given cubic equation must be in the form

$$y^3 + Py^2 + Qy + R = 0 \tag{7-10}$$

and is reduced to the form

$$x^3 + ax + b = 0 \tag{7-11}$$

where

$$a = \frac{1}{3}(3Q - P^2) \tag{7-12}$$

$$b = \frac{1}{27}(2P^3 - 9PQ + 27R) \tag{7-13}$$

$$x = y + \frac{P}{3} \tag{7-14}$$

The smallest root of the original equation becomes

$$y = A + B - \frac{P}{3} \tag{7-15}$$

where

$$A = \left(-\frac{b}{2} + \sqrt{\frac{b^2}{4} + \frac{a^3}{27}}\right)^{1/3} \tag{7-16}$$

$$B = \left(-\frac{b}{2} - \sqrt{\frac{b^2}{4} + \frac{a^3}{27}}\right)^{1/3} \tag{7-17}$$

However, if the term inside the square root portion of A and B is less than or equal to zero, the square root is imaginary and a conjugate imaginary root to the cubic equation is produced. In this case, a trignometric method is necessary to determine the smallest positive real root of the given equation, because an imaginary fluid volume

has no physical significance. The solution obtained from the trigonometric method has the form

$$y = 2\sqrt{\frac{-a}{3}}\cos\left[\frac{1}{3}\cos^{-1}\left(\frac{3b}{a\sqrt{-a/3}}\right) + \frac{2}{3}\pi\right] - \frac{P}{3} \qquad (7\text{-}18)$$

IBMOB1 and IBMOB2

These two routines combine to calculate the interblock mobility ratios for each of the three phases. The equation developed by Stone [33] is used to define the relative permeability to oil as

$$k_{ro} = (k_{row} + k_{rw})(k_{rog} + k_{rg}) - (k_{rw} + k_{rg}) \qquad (7\text{-}19)$$

This procedure incorporates the theory of channel flow in porous media with probability concepts to obtain a means of determining the relative permeability tc oil in the presence of flowing water and gas. No simulation packages in the literature provide a means of allowing for differences in the relative permeability to oil as a result of other flowing phases being present. All of the information required for use in the Stone [33] equation is generally available as a result of reservoir core and fluid property analyses. This method provides a good fit to experimental data except in situations where a high water saturation and low gas saturation are present.

The appropriate phase permeability is modified in IBMOB2 to correspond to an upstream interblock value which is weighted by cell size. The upstream weighting method

is an extrapolation of the phase permeabilities of two upstream cells to the interblock location desired. This value is then used in the interblock mobility equation

$$\lambda_{ji} = \frac{k_{rj}}{\mu_{ji}} \tag{7-20}$$

where i is the cell indicator and j indicates the phase under consideration. Each cell mobility is multiplied at this point by the cell transmissibility value, which is a term consisting of all of the general pressure equation terms which are constant for all cells at any point in time. The transmissibility value is calculated in the mainline of the simulator as

$$Kh_x = \frac{\frac{K_i h_i \Delta y_i}{\Delta x_i} + \frac{K_{i+1} h_{i+1} \Delta y_{i+1}}{\Delta x_{i+1}}}{\Delta x_i + \Delta x_{i+1}} \tag{7-21}$$

$$Kh_y = \frac{\frac{K_i h_i \Delta x_i}{\Delta y_i} + \frac{K_{i+1} h_{i+1} \Delta x_{i+1}}{\Delta y_{i+1}}}{\Delta y_i + \Delta y_{i+1}} \tag{7-22}$$

The mobility value of each cell is replaced by the upstream mobility for the right side boundary (in the x-direction) or the bottom side boundary (in the y-direction) for the given cell. The print indicator for the interblock mobility ratio is checked and if it is any value other than zero, the matrices are printed for examination by the user. Six interblock mobility matrices are printed, each of the three phases (oil, gas, and water) in each of the two directions (x and y).

SATRN

This routine calculates the new phase saturations using the equation developed from the equations describing the reservoir system. Beginning with the phase equations, discussed in Chapter IV, and including the potential equations, we have

$$\nabla[A\rho_j k\left(\frac{k_r}{\mu}\right)_j \nabla(P_j - \rho_j gh)] - q_{vj} = \frac{\partial}{\partial t}(V_k \phi \rho_j S_j) \qquad (7\text{-}23)$$

where j is the phase identifier. Breaking up the right hand side and multiplying the equation by $1/\rho_j$, we obtain

$$\nabla[A\lambda_j \nabla(P_j - \rho_j gh)] - \frac{q_{Vj}}{\rho_j} - \frac{V_k}{\rho_j}[\phi S_j \frac{\partial \rho_j}{\partial t} + \rho_j S_j \frac{\partial \phi}{\partial t}] = V_k \phi \frac{\partial S_j}{\partial t} \qquad (7\text{-}24)$$

where $\lambda$ has been used to replace the interblock mobility ratio term.

Because the updated saturations are required to determine the new overall compositions of each cell, and the updated compositions are necessary to determine the new phase densities, it is assumed that the density change with respect to time is zero and these terms fall out of the above equation. Using finite difference techniques to discretize the new saturation equation, we have

$$S_j^{n+1} = \frac{\Delta t}{V_i \Delta x_i}[A_x \lambda_{xj,i+1}(P_{j,i+1} - P_{j,i}) - A_x \lambda_{xj,i-1}(P_{j,i} - P_{j,i-1})]$$

$$+ \frac{\Delta t}{V_i \Delta y_i}[A_y \lambda_{yj,i+n}(P_{j,i+n} - P_{j,i}) - A_y \lambda_{yj,i-n}(P_{j,i} - P_{j,i-n})]$$

$$- \frac{q_{Vj}\Delta t}{\rho_j V_i} - \frac{S_j^n \Delta t}{\phi^n}[\phi^{n+1} - \phi^n] + S_j^n \qquad (7\text{-}25)$$

This equation is used to find the oil and water saturations of each cell, but the gas saturation is determined from the definition of saturation which says that the three phase saturations must sum to unity. Because of the compressibility of gas mixtures, any error which results from not using the phase saturation for the gas phase is reduced. The mass of the gas phase present in the system is not as dependent on the gas saturation as it is on the system pressure, so a gas saturation is not as critical in the phase material balance check.

UPDATE

This routine updates saturations, compositions, and fluid properties to the new pressure levels obtained from a finite difference solution of the general, oil-phase equation for the reservoir system. In addition, new phase potentials and the new overall composition matrices can be printed if the print identifier for each is some value other than zero.

The updating of the overall compositions is accomplished using finite difference techniques and the relation

$$\nabla\cdot[A\lambda_t\nabla\Phi_\ell]\rho_\ell x_i + \nabla\cdot[A\lambda_t\nabla\Phi_g]\rho_g y_i = V_k \frac{\partial}{\partial t}[\phi(S_\ell\rho_\ell + S_g\rho_g)z_i]$$

The final relation used to determine the new compositions for each cell is presented in Figure 13. These new overall composition values are flashed to determine the individual phase compositions and the new pressure level fluid properties are computed.

Figure 13

Equation for Updated Compositions

$$
\begin{aligned}
Z_i^{n+1} = \frac{1}{S_\ell^{n+1}\rho_\ell^n + S_g^{n+1}\rho_g^n} \Bigg\{ & (S_\ell\rho_\ell + S_g\rho_g)^n Z_i^n \\
& + \frac{\Delta t}{V_i}\Bigg[\frac{A_x\rho_\ell X_i}{\Delta x_i}\left(\lambda_{xt_{i+\frac{1}{2}}}\frac{[(\Phi_\ell)_{i+1} - (\Phi_\ell)_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xt_{i-\frac{1}{2}}}\frac{[(\Phi_\ell)_i - (\Phi_\ell)_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right) \\
& + \frac{A_y\rho_\ell X_i}{\Delta y_i}\left(\lambda_{yt_{i+\frac{1}{2}\eta}}\frac{[(\Phi_\ell)_{i+\eta} - (\Phi_\ell)_i]}{\Delta y_{i+\frac{1}{2}\eta}} - \lambda_{yt_{i-\frac{1}{2}\eta}}\frac{[(\Phi_\ell)_i - (\Phi_\ell)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right) \\
& + \frac{A_x\rho_g Y_i}{\Delta x_i}\left(\lambda_{xt_{i+\frac{1}{2}}}\frac{[(\Phi_g)_{i+1} - (\Phi_g)_i]}{\Delta x_{i+\frac{1}{2}}} - \lambda_{xt_{i-\frac{1}{2}}}\frac{[(\Phi_g)_i - (\Phi_g)_{i-1}]}{\Delta x_{i-\frac{1}{2}}}\right) \\
& + \frac{A_y\rho_g Y_i}{\Delta y_i}\left(\lambda_{yt_{i+\frac{1}{2}\eta}}\frac{[(\Phi_g)_{i+\eta} - (\Phi_g)_i]}{\Delta y_{i+\frac{1}{2}\eta}} - \lambda_{yt_{i-\frac{1}{2}\eta}}\frac{[(\Phi_g)_i - (\Phi_g)_{i-\eta}]}{\Delta y_{i-\frac{1}{2}\eta}}\right)\Bigg]\Bigg\}
\end{aligned}
$$

# CHAPTER VIII

## RESULTS AND RECOMMENDATIONS

In order to verify the simulation package which was developed, a one-dimensional system was studied. The system consists of ten rectangular cells, each of which is one foot by 175 feet and 25 feet thick. Production from the system is out of an end cell (#1) at a flowing pressure of 1500 pounds per square inch. The initial pressure in the system is 2021 psia. The overall hydrocarbon composition is initially as follows:

| | |
|---|---|
| Methane | 0.5043 |
| Ethane | 0.1549 |
| Propane | 0.0875 |
| Iso-Butane | 0.0236 |
| N-Butane | 0.0353 |
| Pentane | 0.0295 |
| Hexane | 0.0229 |
| Heptane | 0.1420 |

The time step size chosen for this study is 0.05 days, and the system response was studied for 10.0 days. The pressure distribution obtained during this simulation study is presented in Table 4.

TABLE 4

Pressure Response of Test Case

| Time | Cell #1 | Cell #2 | Cell #3 | Cell #4 |
|---|---|---|---|---|
| 0.0 | 2021. | 2021. | 2021. | 2021. |
| 0.5 | 1775. | 2016. | 2021. | 2021. |
| 1.0 | 1735. | 2007. | 2021. | 2021. |
| 1.5 | 1716. | 1996. | 2020. | 2021. |
| 2.0 | 1708. | 1985. | 2020. | 2021. |
| 2.5 | 1705. | 1974. | 2019. | 2021. |
| 3.0 | 1706. | 1964. | 2018. | 2021. |
| 3.5 | 1708. | 1954. | 2017. | 2021. |
| 4.0 | 1713. | 1945. | 2016. | 2021. |
| 4.5 | 1717. | 1937. | 2015. | 2021. |
| 5.0 | 1723. | 1930. | 2014. | 2021. |
| 5.5 | 1729. | 1923. | 2012. | 2020. |
| 6.0 | 1734. | 1917. | 2011. | 2020. |
| 6.5 | 1740. | 1911. | 2009. | 2020. |
| 7.0 | 1746. | 1906. | 2008. | 2020. |
| 7.5 | 1751. | 1902. | 2006. | 2020. |
| 8.0 | 1756. | 1898. | 2004. | 2020. |
| 8.5 | 1761. | 1894. | 2002. | 2019. |
| 9.0 | 1766. | 1891. | 2001. | 2019. |
| 9.5 | 1770. | 1888. | 1999. | 2019. |
| 10.0 | 1775. | 1886. | 1997. | 2018. |

As can be seen by the pressure distributions, the simulation produces results which are reasonable based on our experience. As fluids are produced out of the end cell of the system, the pressure within that cell decreases. This produces a response in the next cell when the pressure drop across the boundary is sufficiently high to cause flow between the cells.

For testing purposes, it is assumed that only the oil phase was being produced from the reservoir. This results in a decrease in the oil saturation for the producing cell which in turn leads to a decrease in the relative permeability to oil and a decrease in oil production from the cell. The production eventually decreases to the point that influx into the cell continues at a greater rate than oil is being produced. A less mobile oil phase is also entering the cell but at a rate less than the production demand. The cell pressure increases due to the increasing influx of fluid; however, oil production continues to decrease. This explains the increase in pressure in cell one beginning at [illegible] days. The other cell pressures continue to decline as fluids move to the low pressure region.

Several weaknesses were found in the simulation package during the verification process. These will be discussed more fully.

K-Value Algorithm

The K-value algorithm used in this simulation package is felt to be a potential source of error. The deviations

evidenced during the developmental investigation also appear in a comparison of NGPSA K-values to those calculated for the test case at initial conditions:

| NGPSA | Test Case |
|---|---|
| 2.0 | 1.55 |
| 1.08 | 1.11 |
| .88 | .77 |
| .64 | .63 |
| .58 | .59 |
| .40 | .46 |
| .25 | .41 |
| .16 | .37 |

The K-value algorithm should be the area of primary concern in future studies based on this work. A relatively simple, yet very accurate, correlation must be used. A great deal of multicomponent real system data should be gathered and fit to a function of the critical properties of the individual components, the convergence pressure, and the system pressure and temperature. This data should primarily cover the range of pressures and compositions typically encountered in a petroleum reservoir in order to provide a correlation which is most accurate in the range needed. Special attention should be paid to accuracy and efficiency. An improved algorithm can be easily inserted in place of the current procedure.

Injection and Production

A second weakness in this package is the subroutine which determines the injection or production rates. The production rates computed for the one-dimensional system under study were larger than expected. This possible error should be studied, and it would be valuable to include a procedure to determine the compositions of each phase from the surface separators, taking into account the continuous flashing as a result of decreases in pressure and temperature as the fluid moves up the wellbore.

Fluid Properties

The viscosity and density correlations do not appear to cause significant errors in this particular study. In the area of the critical point of a given system these correlations deviate from reality. At the critical point of normal butane, for example, the viscosity and density correlations produce the following results:

| | Liquid | Vapor |
|---|---|---|
| Density (lb./ft$^3$) | 57.6 | 1.059 |
| Viscosity (cp) | 141.5 | 0.011 |

The liquid and vapor phase properties should be identical at the critical point. Although these results indicate a very large error at the critical point, accuracy figures presented by the authors of the various procedures show that the correlations predict real behavior very well in the wide temperature and pressure regions encountered in a petroleum reservoir. The large divergence from real behavior at the

critical point is a regional error and probably significant only at system conditions within a few units of the critical point. Inaccuracies in the region of the critical point are not intolerable because of the accuracy displayed in other regions, as long as the package user is aware of this possible source of error.

Improvements in the viscosity and density algorithms will result in a more flexible system. Before such improvements are made, the accuracy of these algorithms should be more thoroughly studied to determine if the region of error is large enough to justify the effort involved in improvements.

## Conclusions

1. Of the four different types of petroleum fluids which have been discussed, the gas condensate and volatile oil reservoirs experience significant phase change effects during depletion. The black oil model, which lacks the essential features to handle transfer of mass between phases, is inadequate for predicting the behavior of these systems. A compositional approach is necessary to accurately predict the performance of petroleum reservoir systems which exhibit significant interphase mass transfer.

2. Fluid flow in porous media and thermodynamic equilibrium behavior of reservoir fluids are coupled to produce a workable mathematical model for compositional systems. This mathematical model is verified for the one-dimensional case.

3. Versatility in the numerical techniques utilized in the mathematical model is required to accommodate the different reservoir geometries and well configurations encountered in petroleum

reservoir systems. The simulation package developed incorporates this versatility by providing the option of utilizing either the Alternating Direction Implicit Procedure, the Iterative Alternating Direction Implicit Procedure, or the Line Successive Over-Relaxation Precedure to solve the reservoir system equation.

4. A prominent conclusion of this research is the indication that fast and accurate correlations for the equilibrium K-value and the system fluid properties are required to expedite the compositional simulation.

5. Efforts to minimize computer storage have produced a model which requires less than 200,000 bytes of storage in order to handle the compositional reservoir simulation of a 500-cell, ten component system. The literature provides no information on the computer storage required for similar packages.

6. Although the simulation package developed is more complex mathematically, it displays a computational superiority to other packages discussed in the literature. MacDonald [8] uses a CDC 6600 computer and reports execution times of 0.02 second/cell/component/iteration for a one-dimensional system. In contrast, this package, using an IBM 370/158 computer system, performed a one-dimensional study which required 0.002 seconds/cell/component/iteration.

In this research, a model has been developed in which capillary forces have been considered and three-phase flow permeabilities have been included to provide a prediction method which closely approaches real reservoir behavior. Storage and computational considerations have produced an economically competitive package.

# APPENDICES

## APPENDIX A

### Nomenclature

| | |
|---|---|
| A,B,C,D | elements of tridiagonal matrix system from finite difference techniques |
| $A_i$ | cross-sectional area to flow in i'th direction |
| $A_{mn}$ | element of 6 * 6 matrix of coefficients for vapor density correlation |
| B | formation volume factor of given phase |
| f(L) | flash function evaluated at overall liquid phase mole fraction |
| f'(L) | derivative of flash function |
| g | gravity |
| h | thickness of cell |
| $H_k$ | acceleration parameter for IADIP |
| k | absolute permeability |
| $k_r$ | relative permeability to given phase |
| $K_i$ | equilibrium constant for i'th component in vapor-liquid system |
| L | mole fraction of system in liquid phase |
| $M_{AV}$ | system average molecular weight |
| $M_i$ | molecular weight of i'th component |
| P | system pressure |
| $P_{cgo}$ | capillary pressure to oil |
| $P_{cwo}$ | capillary pressure to water |

| | |
|---|---|
| $P_{conv}$ | convergence pressure of given system |
| $P_c$ | critical pressure |
| $P_i$ | pressure of i'th cell |
| $P_j$ | pressure of given phase |
| $P_m$ | Legendre polynomial of degree 0 through 5 |
| $P_R$ | reduced pressure |
| $P_{RS}$ | reduced component vapor pressure |
| $P_{wf}$ | flowing bottom hole pressure of well |
| q | phase flow rate as production or injection |
| R | universal gas constant |
| S | skin effect due to wellbore damage |
| S | phase saturation of phase of interest |
| t | time |
| T | system temperature |
| $T_c$ | critical temperature |
| $T_R$ | reduced temperature |
| $V_i$ | pore volume of i'th cell |
| $V_k$ | bulk volume of cell |
| V | volume of liquid or vapor system |
| V | mole fraction of system in vapor phase |
| $W_i$ | weight fraction of i'th component |
| $\Delta x$ | x dimension of the given cell |
| $x_i$ | mole fraction of i'th component in liquid phase |
| $\Delta y$ | y dimension of the system |
| $y_i$ | mole fraction of i'th component in vapor phase |
| $z_i$ | overall mole fraction of i'th component present in vapor-liquid system |

| | |
|---|---|
| $z_c$ | critical compressibility of i'th component |
| $\theta$ | Crank-Nicholson parameter (= 0 for fully explicit, = 1 for fully implicit) |
| $\phi$ | rock porosity |
| $\eta$ | number of x-direction columns in system |
| $\Phi$ | flow potential term (except in reference to convergence pressure) |
| $\rho_i$ | density at standard conditions of i'th component |
| $\lambda$ | interblock mobility term for each phase |
| $\mu_i$ | low pressure, pure gas viscosity |
| $\tau$ | dummy factor in viscosity correlations |
| $\mu_m$ | low pressure gas mixture viscosity |
| $\mu$ | phase viscosity |

Subscripts

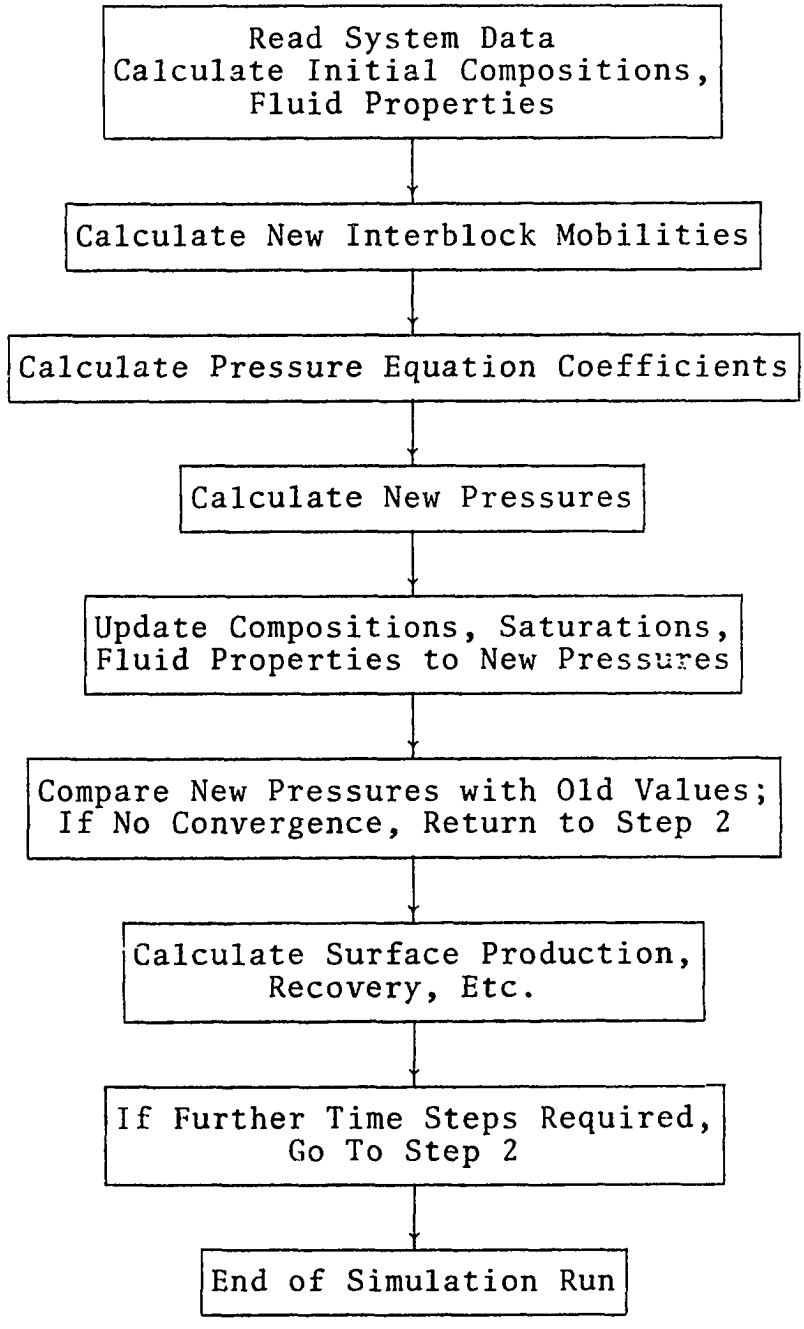
| | |
|---|---|
| i | component or cell identifier |
| j | phase identifier (oil, gas, or water) |
| k | iteration counter |
| L | value for liquid phase |
| m | mixture value |
| V | value for vapor phase |
| C7+ | value for heptanes-plus fraction |

# APPENDIX B

## Flow Chart of Simulator

The simplified flow chart for the simulation package developed is presented on the following page.

Simplified Compositional Simulator Flow Chart

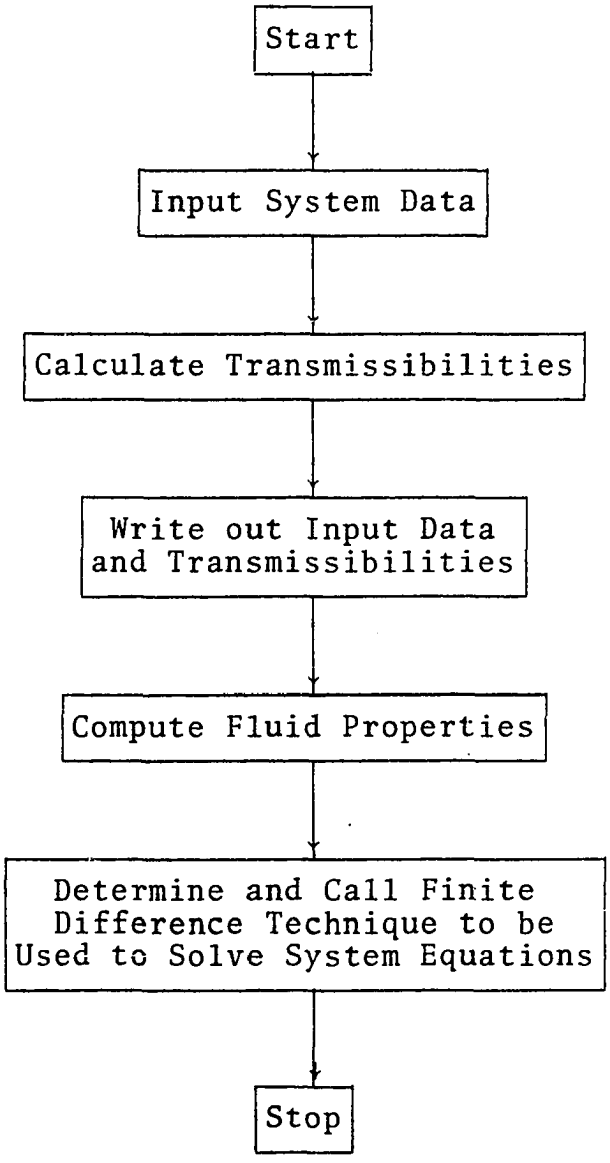


# APPENDIX C

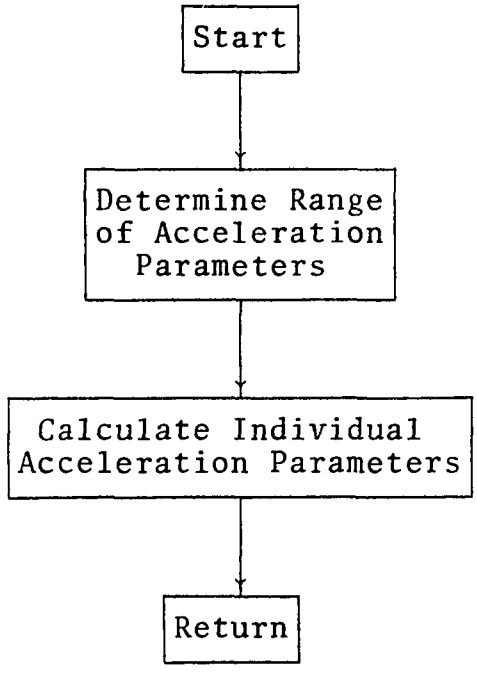
## Flow Charts of Individual Subroutines

Flow charts of each of the subroutines used in this reservoir simulation package are presented on the following pages.

Flow Chart for Mainline



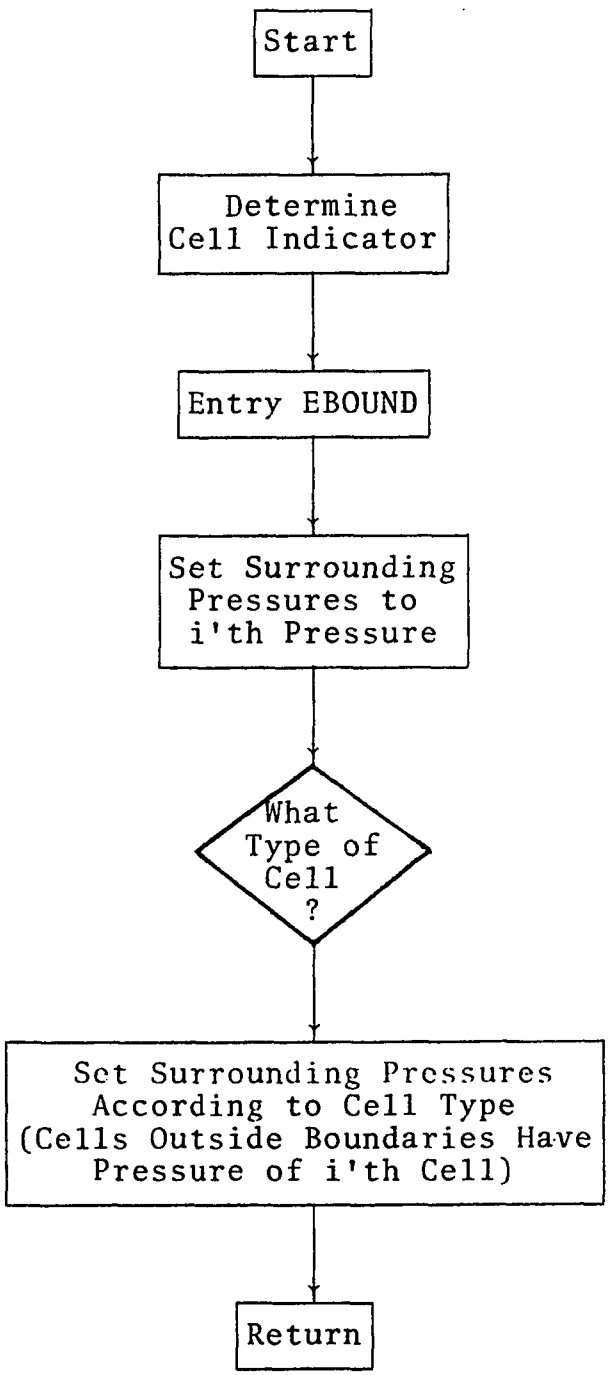
Flow Chart for Subroutine ACCP




Purpose of Subroutine:

To determine the acceleration parameter values to be used in the Iterative Alternating Direction Procedure when none are specified by the package user.

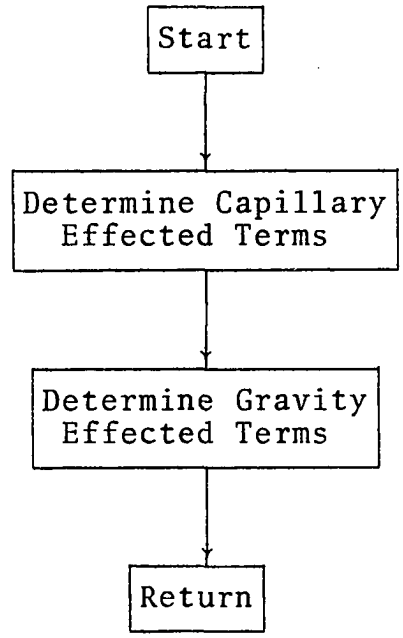
Flow Chart for Subroutine BOUND




Purpose of Subroutine:

To determine values to be used for surrounding cells based on their position in the model.

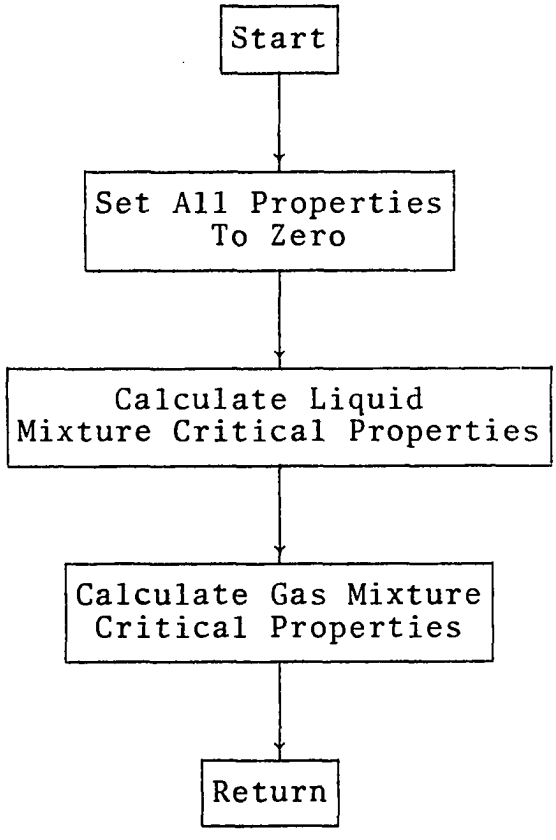
Flow Chart for Subroutine CPGRV




Purpose of Subroutine:

To determine the values of the capillary and gravity terms of the finite difference equations.

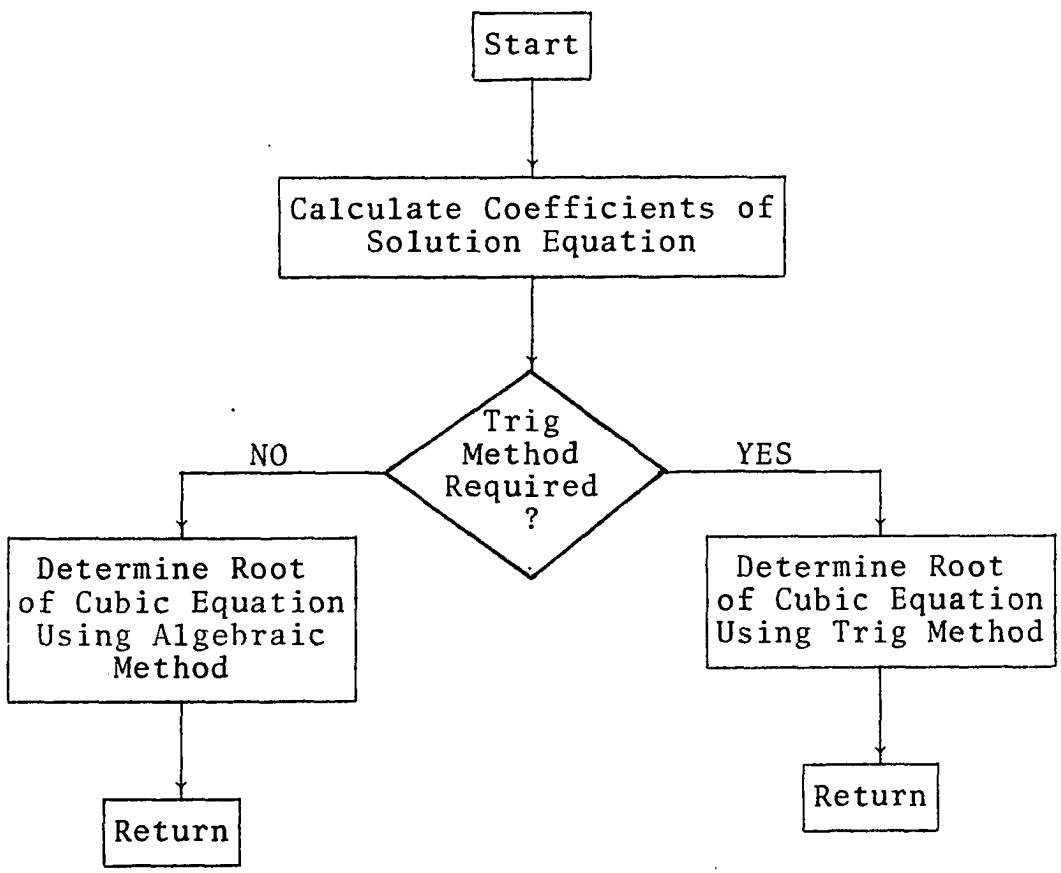
Flow Chart for Subroutine CRTPRP




Purpose of Subroutine:

To determine the liquid and vapor critical properties for a given hydrocarbon system.

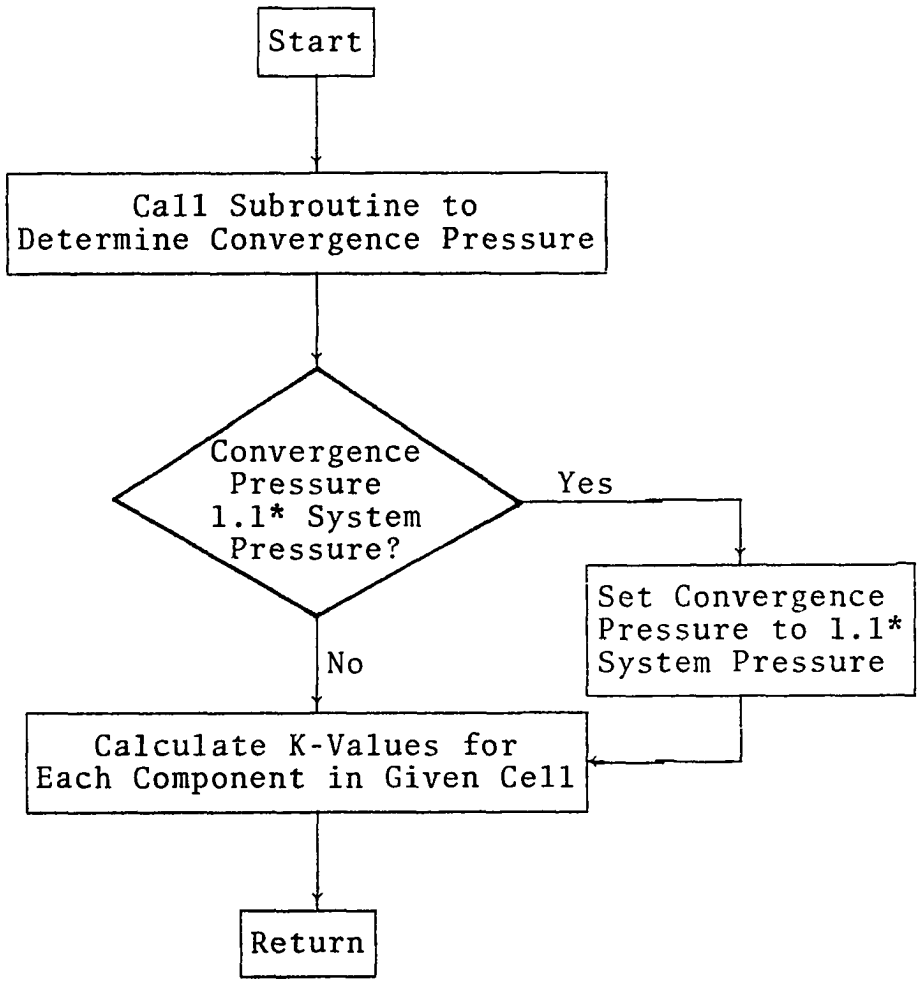
Flow Chart for Function CUBEQN




Purpose of Subroutine:

To determine the smallest positive root of a cubic equation.

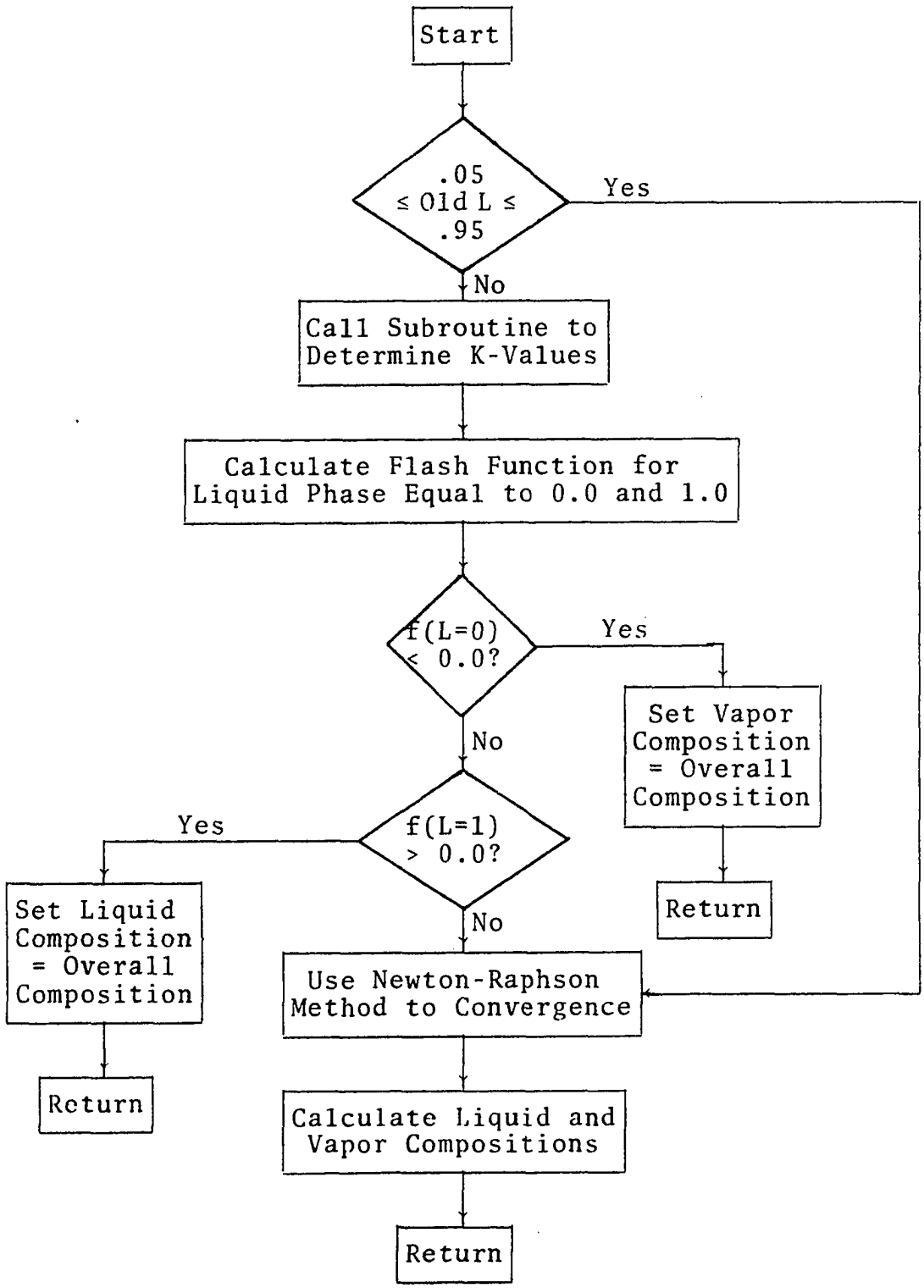
Flow Chart for Subroutine EQUILK




Purpose of Subroutine:

To determine the equilibrium K-Values for a given system.

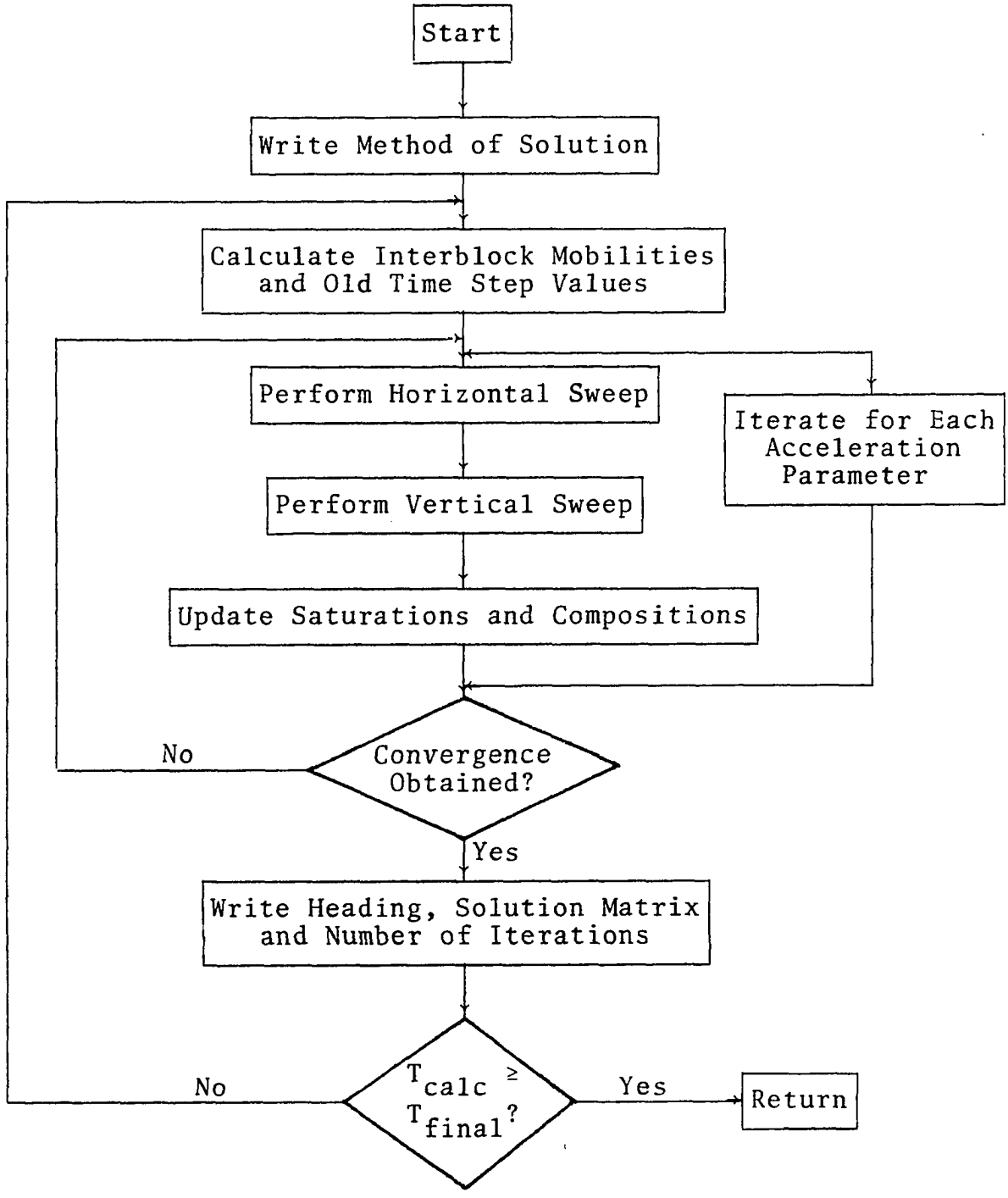
Flow Chart for Subroutine FLASH




Purpose of Subroutine:

To perform a flash calculation on a given fluid system.

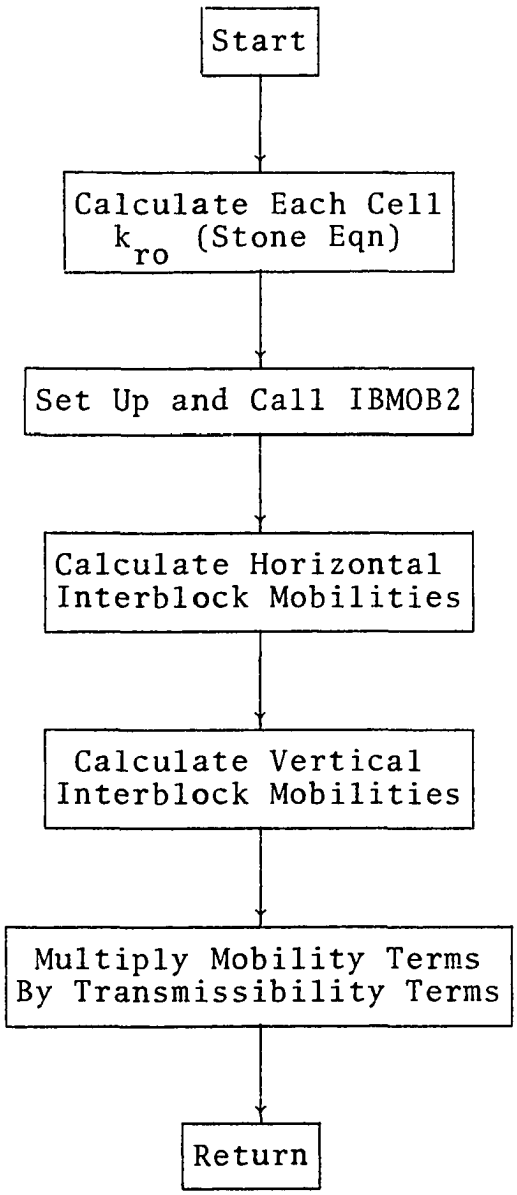
Flow Chart for Subroutine IADIP




Purpose of Subroutine:

To perform the iterative or non-iterative Alternating Direction Implicit Procedure on a given grid system.

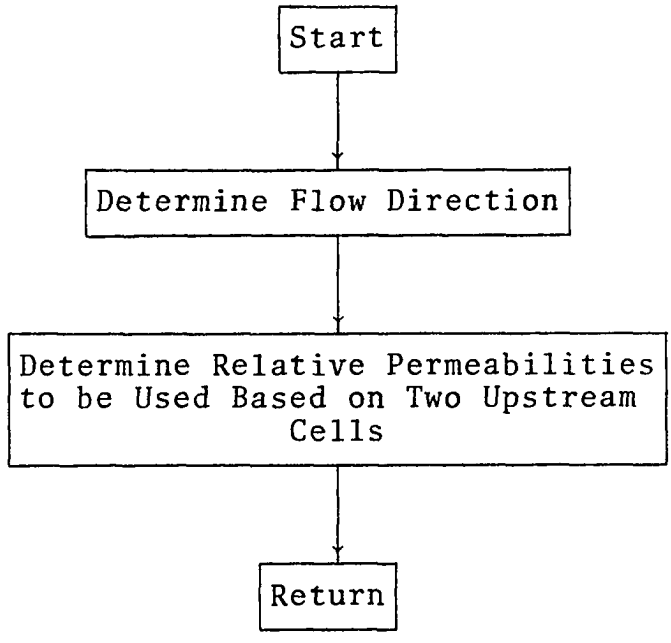
Flow Chart for Subroutine IBMOB1




Purpose of Subroutine:

To determine the interblock mobility terms for a given system.

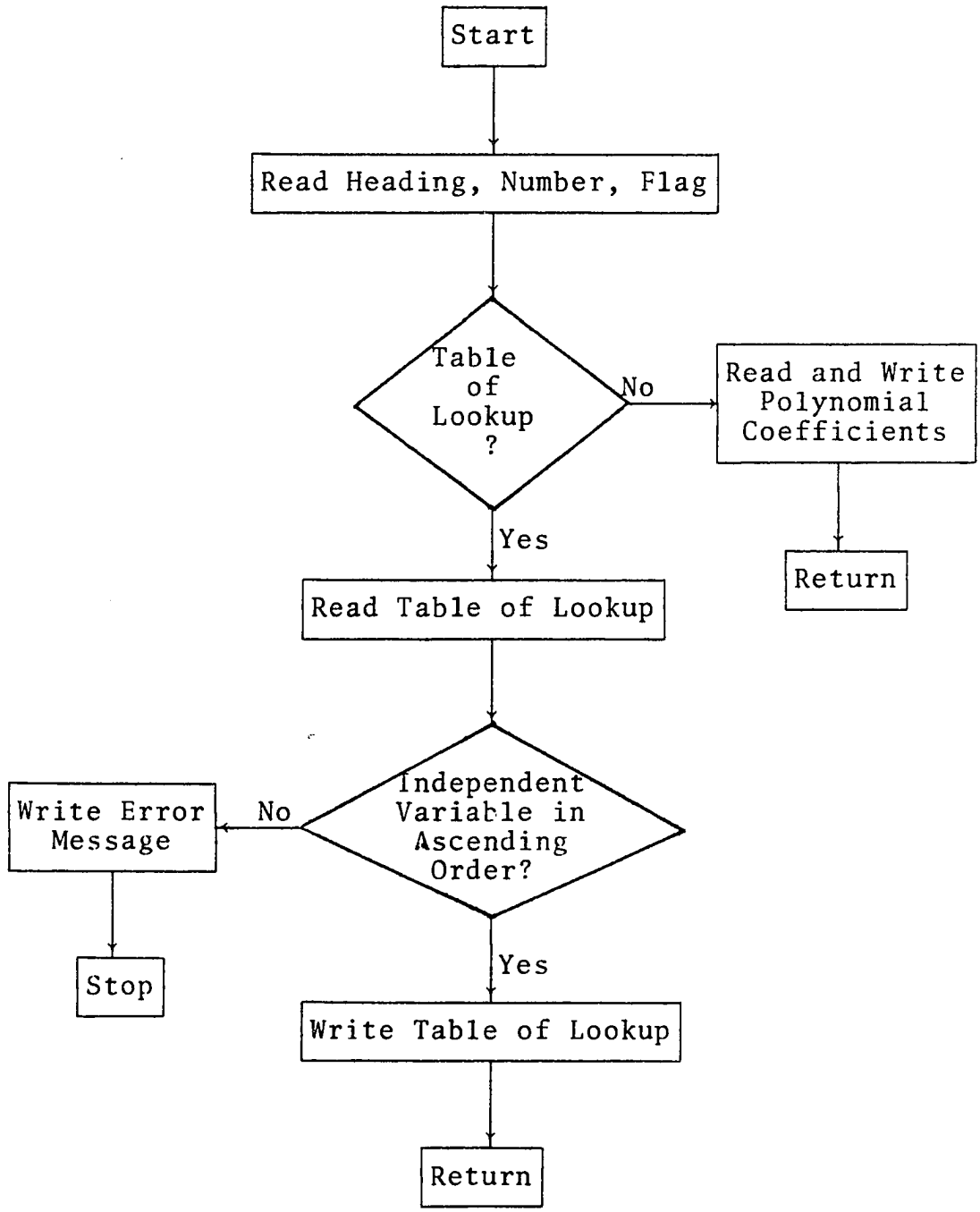
Flow Chart for Subroutine IBMOB2




Purpose of Subroutine:

To supplement IBMOB1 in determining the interblock mobilities.

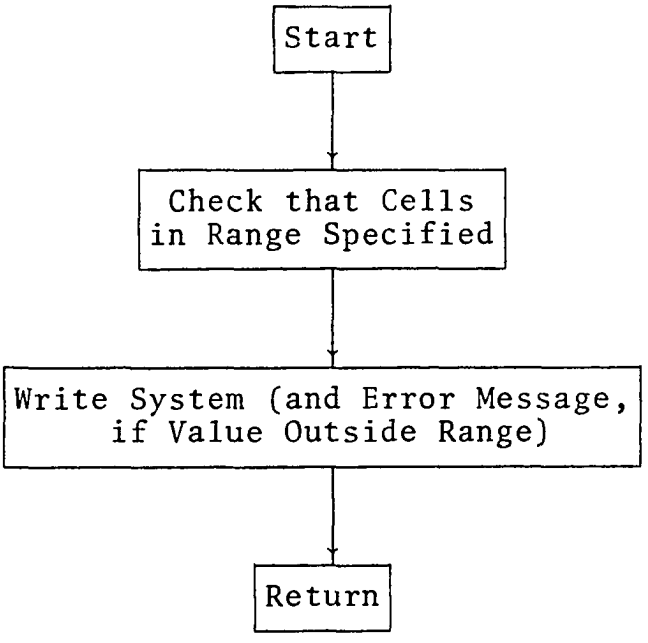
Flow Chart for Subroutine INDATA




Purpose of Subroutine:

To read in data as a table of lookup or polynomial coefficients.

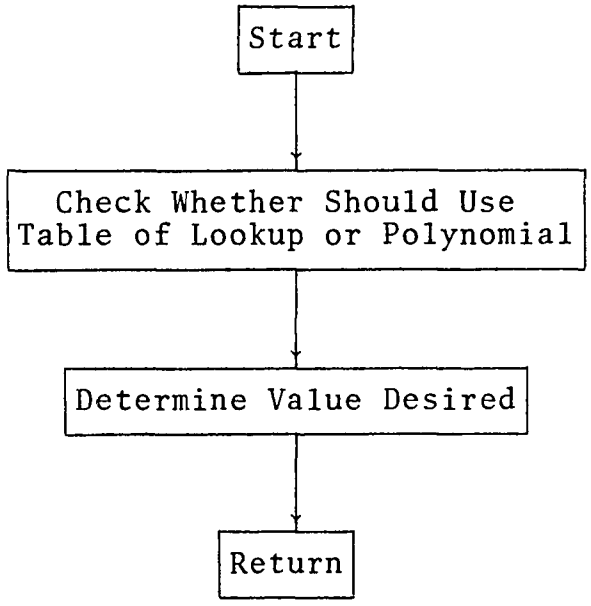
Flow Chart for Subroutine INPUT




Purpose of Subroutine:

To read data matrices, assigning a value to each cell, and check that data lie within range specified.

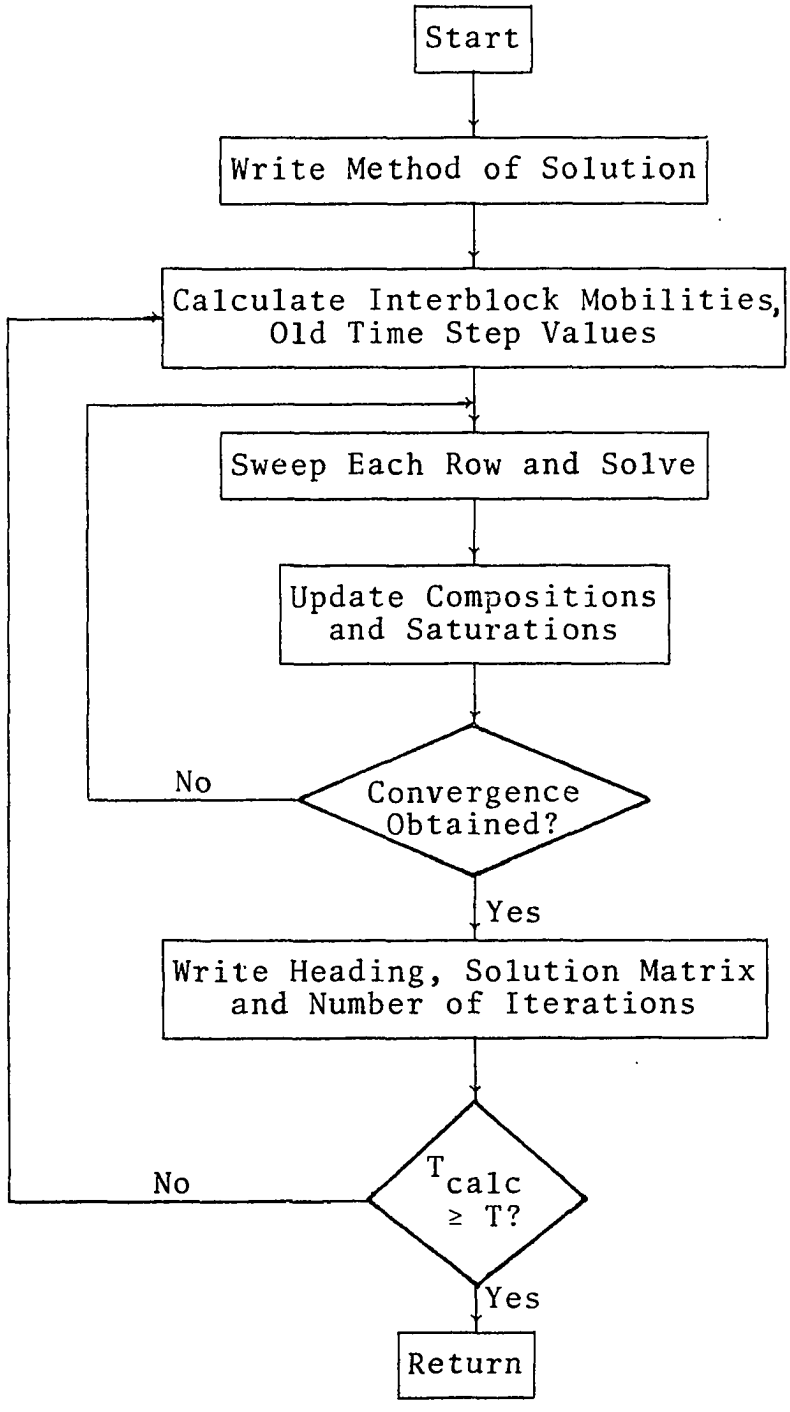
Flow Chart for Subroutine LOOKUP




Purpose of Subroutine:

To use table of lookup data or polynomial curve fit data to determine dependent value from given independent value.

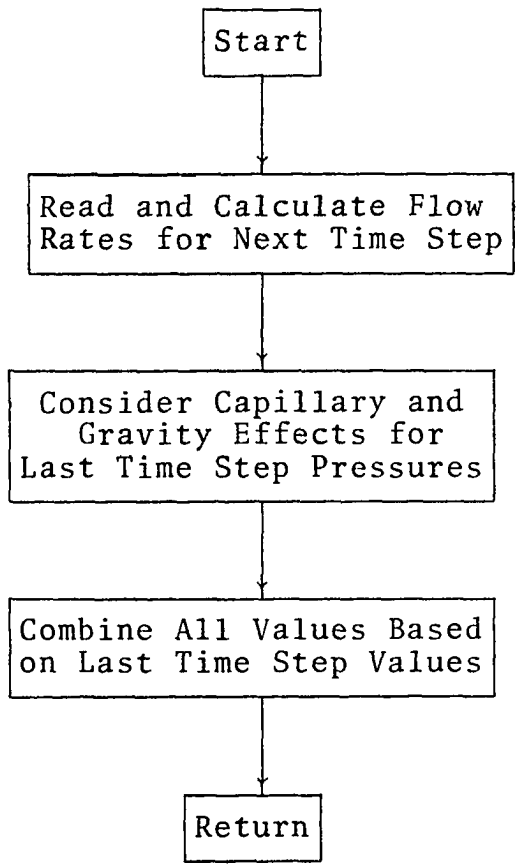
Flow Chart for Subroutine LSOR




Purpose of Subroutine:

To perform the Line Successive Over-Relaxation Procedure on a given system.

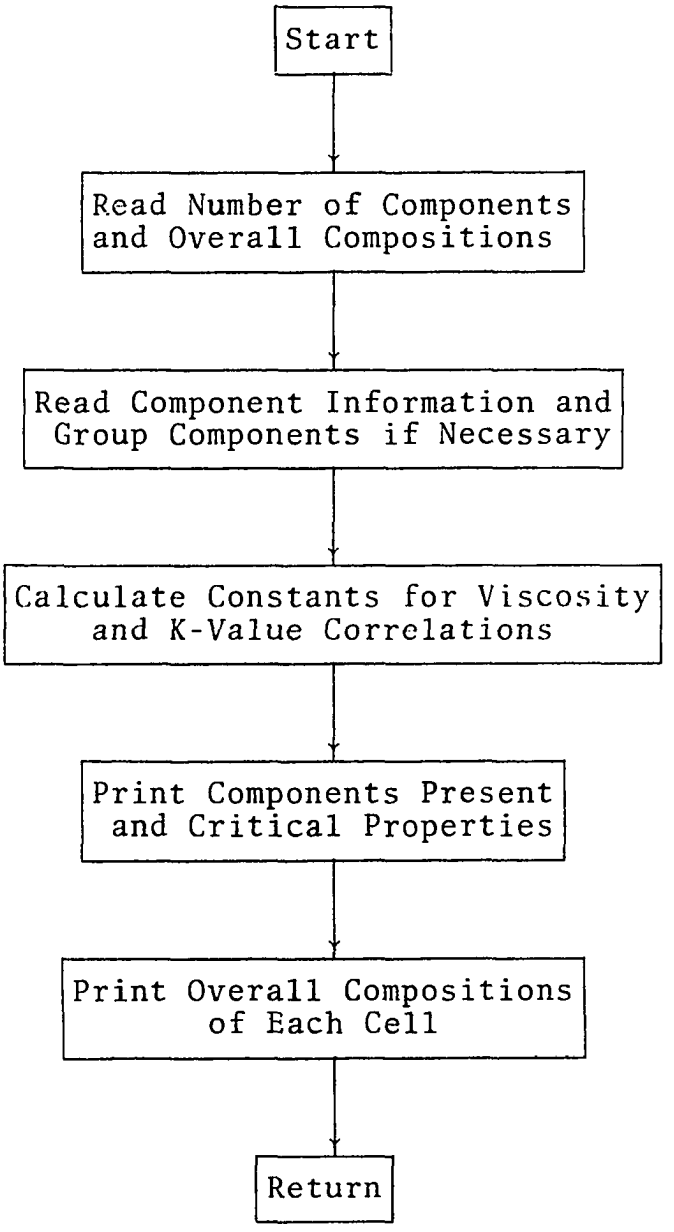
Flow Chart for Subroutine OLDT




Purpose of Subroutine:

To calculate all values based on values from last time step.

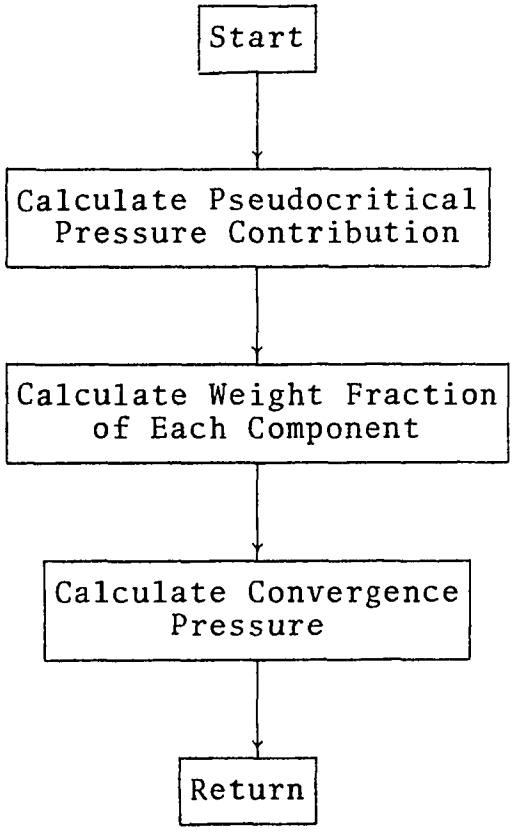
Flow Chart for Subroutine ONCE




Purpose of Subroutine:

To read component data and perform all of the one-time calculations for fluid properties.

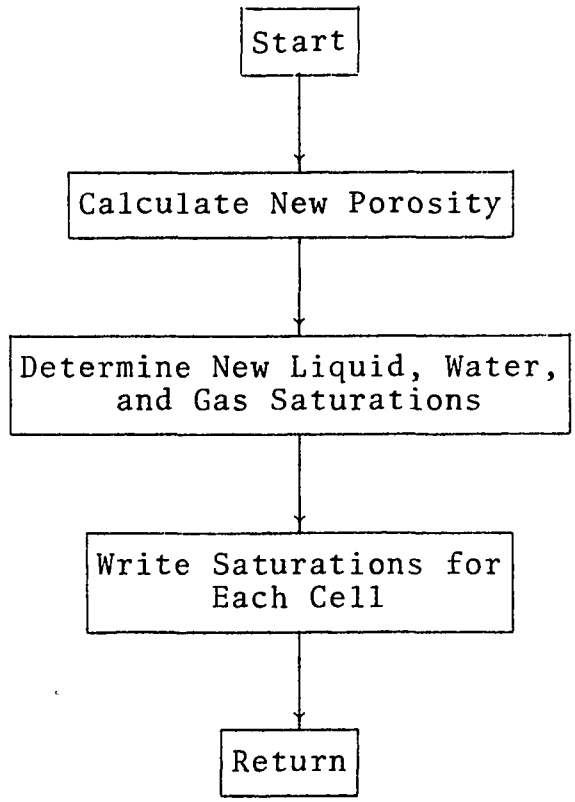
Flow Chart for Subroutine PCONV




Purpose of Subroutine:

To determine the convergence pressure for a given system.

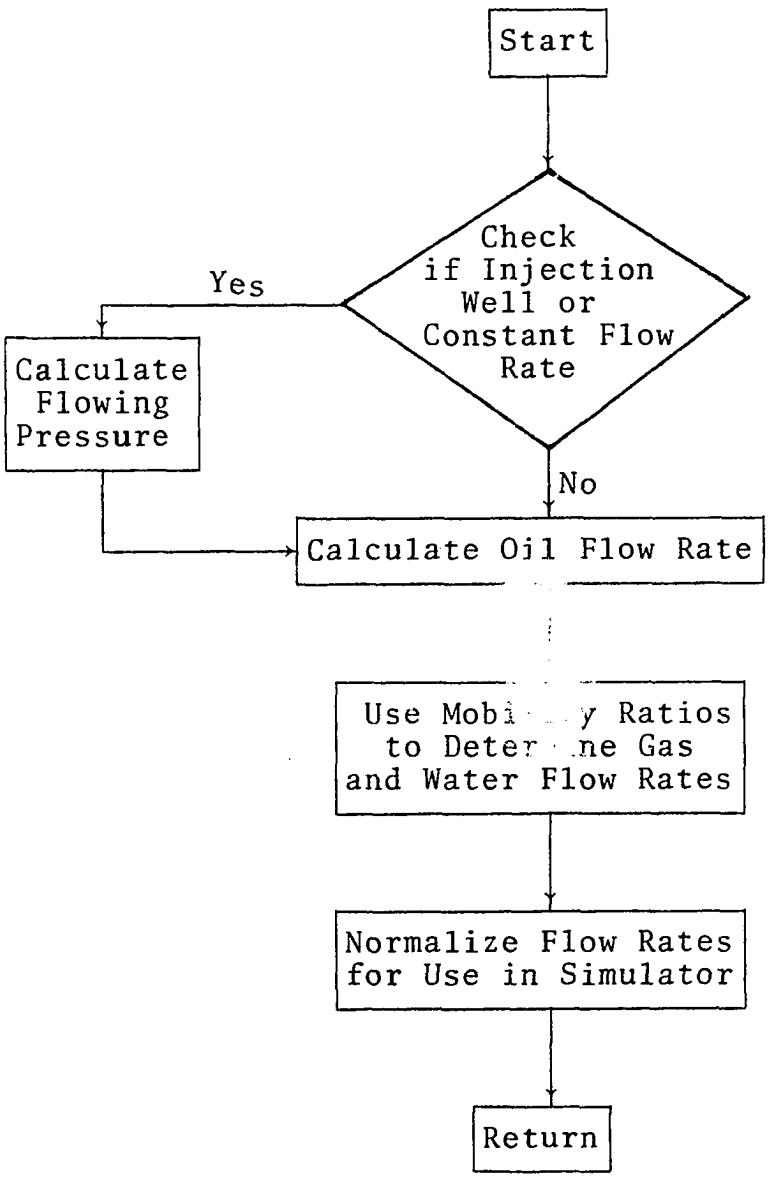
Flow Chart for Subroutine SATRN




Purpose of Subroutine:

To determine the liquid, gas, and water saturations for each cell of system.

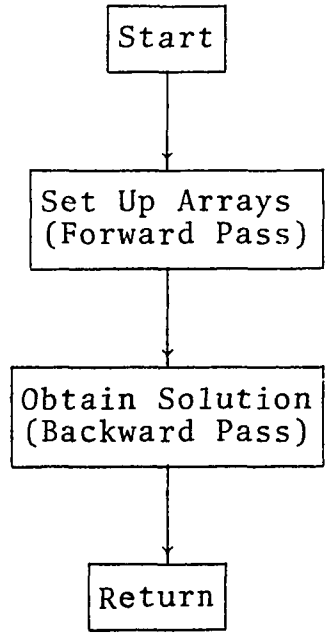
Flow Chart for Subroutine SORSNK




Purpose of Subroutine:

To calculate flow rates and flowing pressures for injector and producer cells.

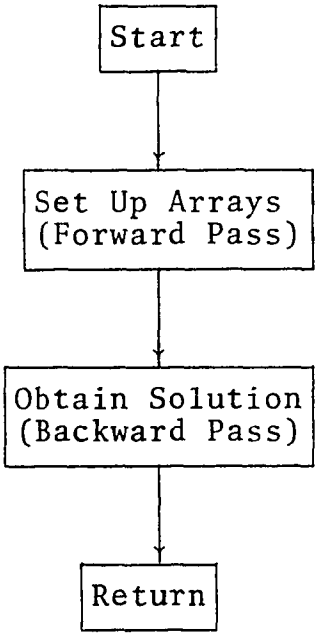
Flow Chart for Subroutine TRIHOR




Purpose of Subroutine:

To perform the horizontal version of the Thomas Algorithm for a close band tridiagonal matrix.

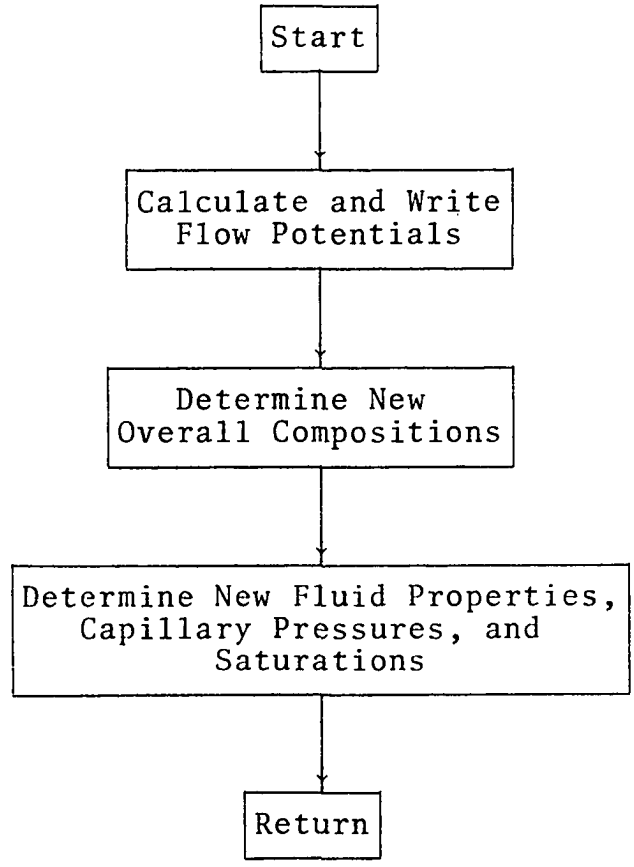
Flow Chart for Subroutine TRIVRT




Purpose of Subroutine:

To perform the vertical version of the Thomas Algorithm for a wide band tridiagonal matrix.

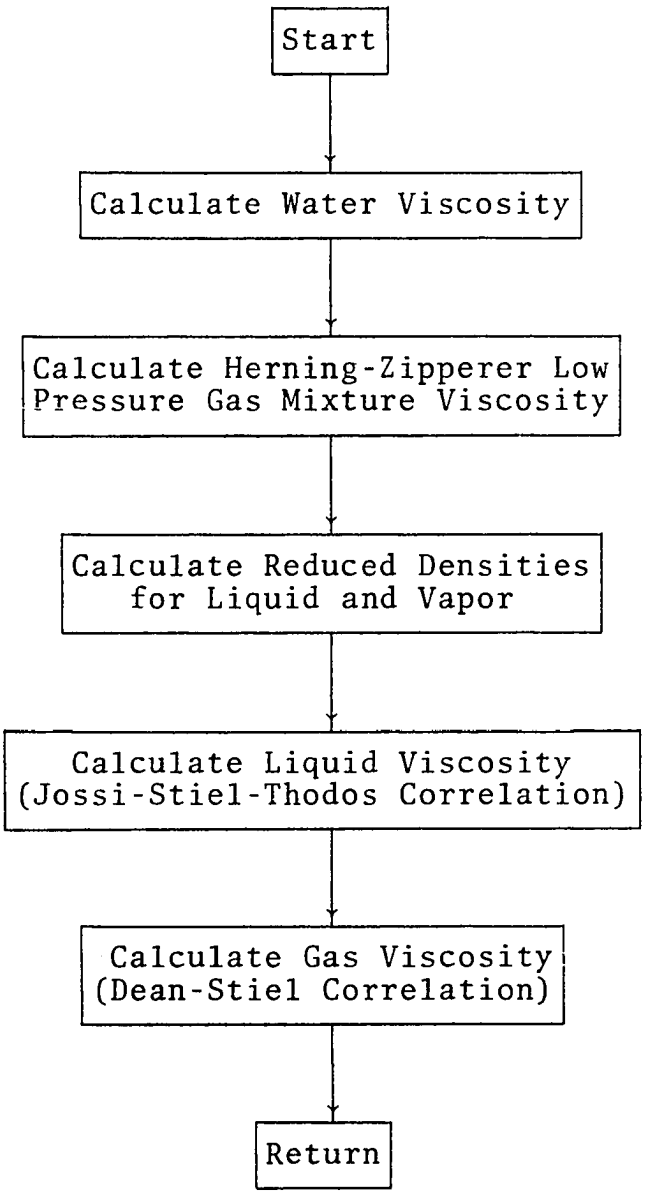
Flow Chart for Subroutine UPDATE




Purpose of Subroutine:

To update all values to the new pressure levels.

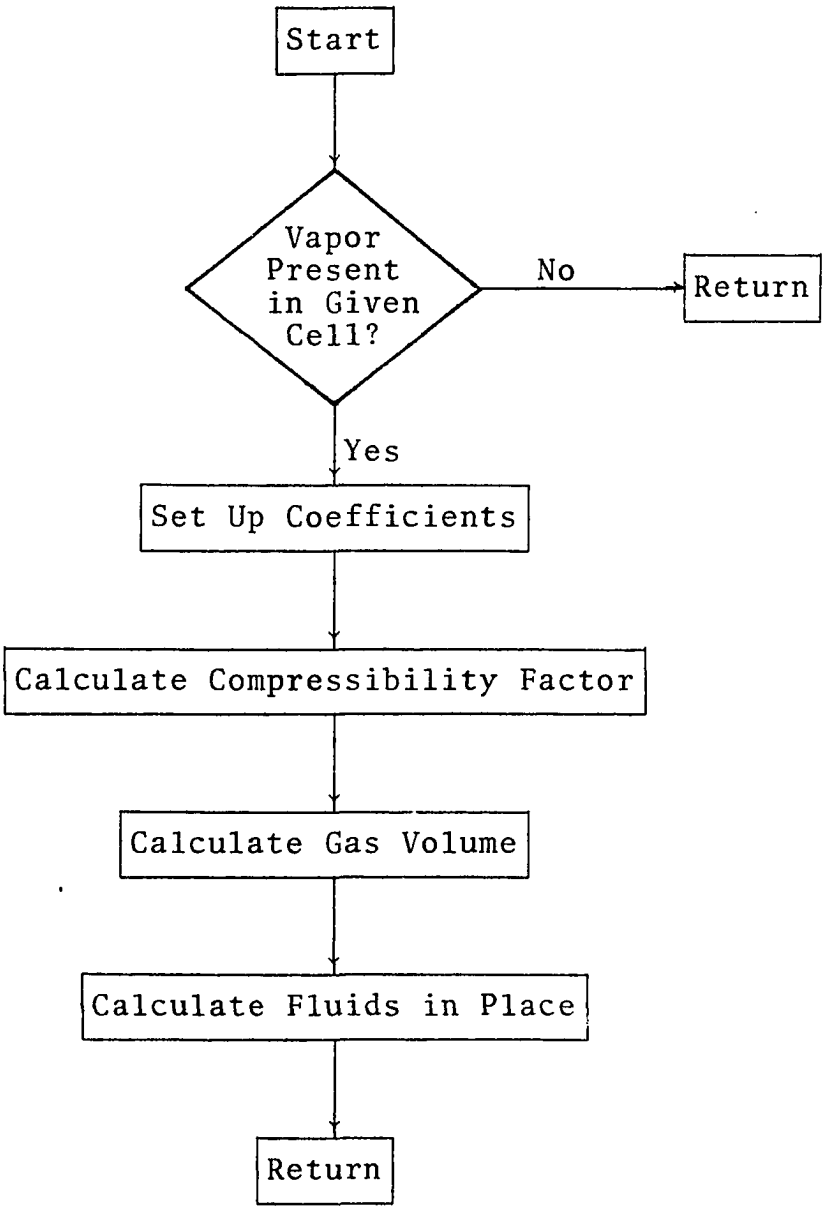
Flow Chart for Subroutine VISCTY




Purpose of Subroutine:

To determine each of the phase viscosities for a given system.

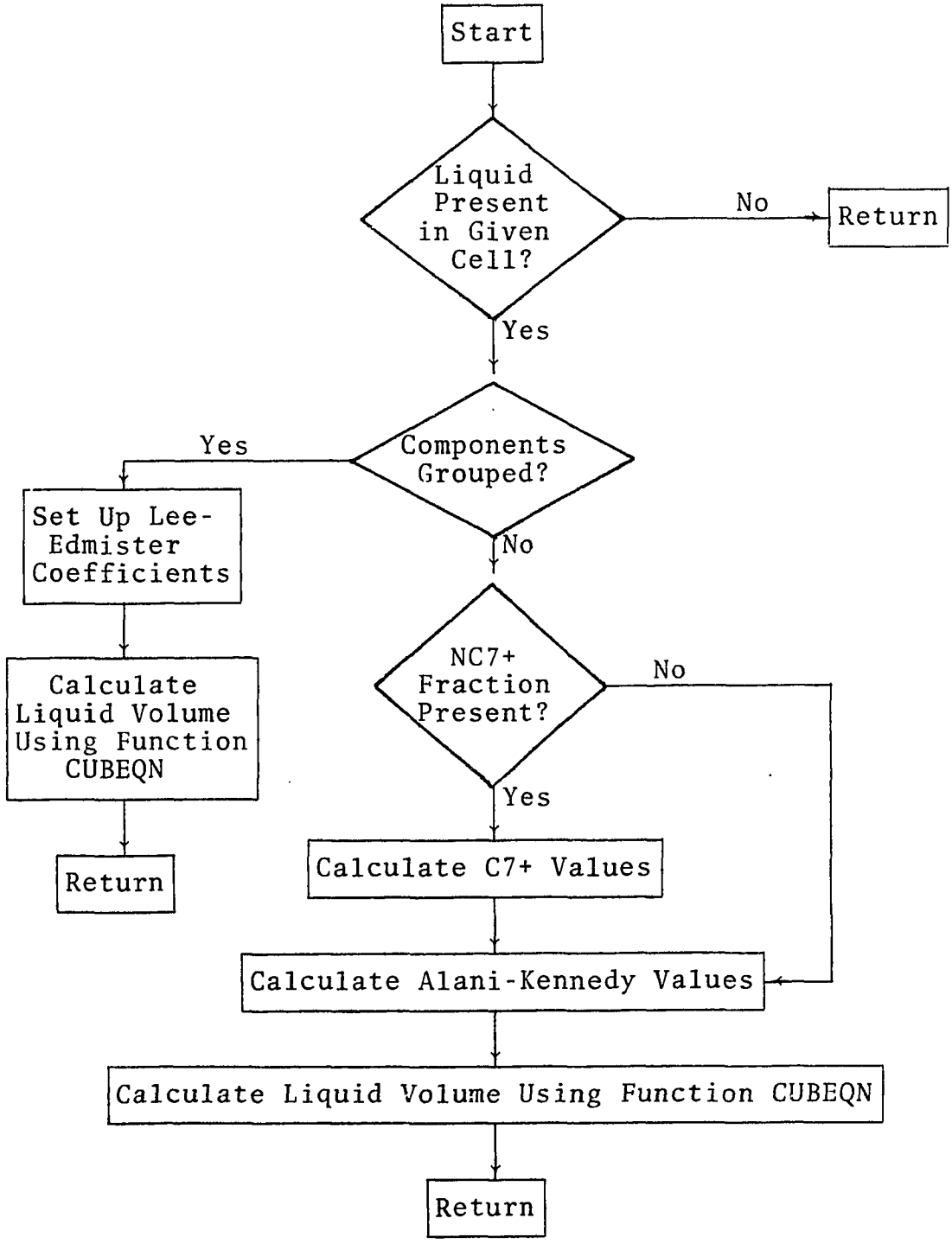
Flow Chart for Subroutine VOLGAS




Purpose of Subroutine:

To determine the vapor volume for a given system.

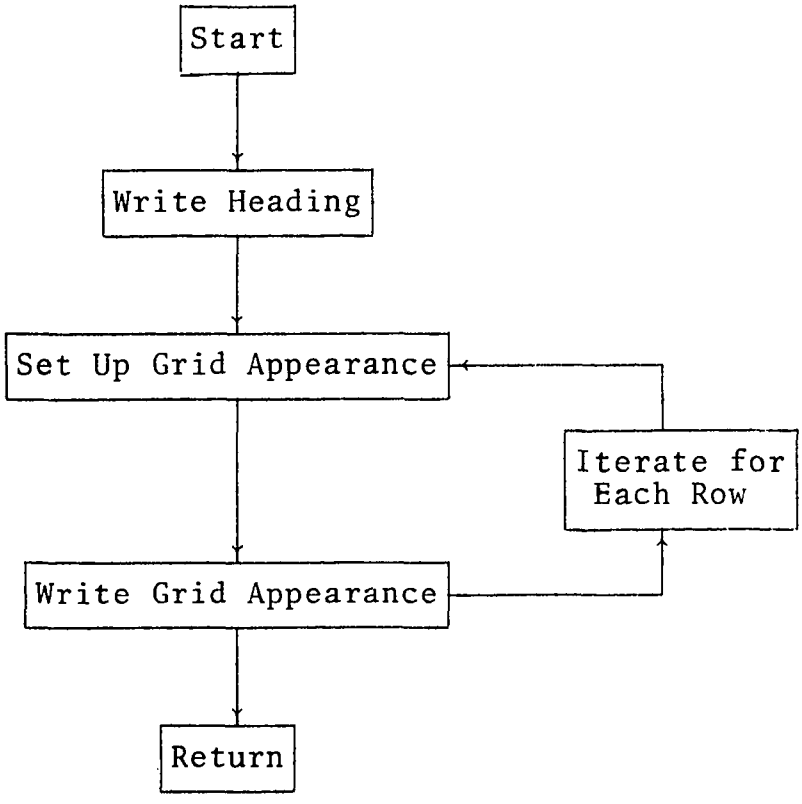
Flow Chart for Subroutine VOLLIQ




Purpose of Subroutine:

To determine the liquid volume for a given system.

Flow Chart for Subroutine WRTMOD




Purpose of Subroutine:

To output a representation of the grid system under consideration.

# APPENDIX D

## FORTRAN Listings

The FORTRAN listing for the simulation package developed is presented on the following pages.

```
C   THIS PROGRAM CANNOT BE RUN CONSECUTIVELY FOR SEVERAL DIFFERENT          00000010
C     SYSTEM CONFIGURATIONS; DEAD CELL VALUES MUST BE REZEROED FOR          00000020
C     DIFFERENT CONFIGURATIONS.                                             00000030
C                                                                           00000040
C                                                                           00000050
      BLOCK DATA                                                            00000060
C   PURPOSE OF SUBPROGRAM:                                                  00000070
C     TO INITIALIZE VARIOUS VALUES FOR USE IN THE SIMULATOR                 00000080
      COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                     00000090
      COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                     00000100
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                00000110
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                 00000120
      COMMON/RPERM/TKRO(500),TKRW(500),TKRG(500)                            00000130
      COMMON/VSCTY/UMO(500),UMG(500),UMW(500)                               00000140
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW       00000150
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                           00000160
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                        00000170
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP             00000180
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                  00000190
      COMMON/SATN/SW(500),SG(500),SO(500),CF                                00000200
      COMMON/TKH/TKHX(500),TKHY(500)                                        00000210
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                              00000220
      DATA IN,IO/5,6/                                                       00000230
      DATA GH,VOL,B,TKRO,UMO,UMG,UMW/3500*0.0/                              00000240
      DATA TLAMOX,TLAMOY,TLAMGX,TLAMGY,TLAMWX,TLAMWY/3000*0.0/              00000250
      DATA DL,DG,DW,XL,ICN,NPQ/1560*0.0,80*0/                               00000260
      DATA TKHX,TKHY,PCWO,PCGO,SO,SG,SW/3500*0.0/                           00000270
      END                                                                   00000280
C                                                                           00000290
C                                                                           00000300
C   PURPOSE OF MAINLINE:                                                    00000310
C     TO INPUT ALL OF THE DATA REQUIRED TO DEFINE THE GRID                  00000320
C       SYSTEM AND TO PERFORM SOME OF THE ONE TIME CALCULATIONS             00000330
      COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                       00000340
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                              00000350
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                  00000360
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW       00000370
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                 00000380
      COMMON/TKH/TKHX(500),TKHY(500)                                        00000390
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                00000400
      COMMON/SATN/SW(500),SG(500),SO(500),CF                                00000410
      COMMON/FLUID3/AKROG(10),AKRCW(10),AKRG(10),AKRW(10)                   00000420
      COMMON/FLUID4/SKROG(10),SKROW(10),SKRG(10),SKRW(10)                   00000430
      COMMON/PERMF/NKROG,IFKROG,NKROW,IFKROW,NKRG,IFKRG,NKRW,IFKRW          00000440
      COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                     00000450
```

```
      COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                    00000460
      COMMON/RPERM/TKRO(500),TKRW(500),TKRG(500)                           00000470
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                       00000480
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP            00000490
      COMMON/WTRV/AUW(10),PUW(10),NUW,IFUW                                 00000500
      COMMON/FVOLF/ABO(10),PBO(10),NBO,IFBO,ABW(10),PBW(10),NBW,IFBW       00000510
      COMMON/PCAPG/APCG(10),PPCG(10),NPCG,IFPCG                            00000520
      COMMON/PCAPO/APCO(10),PPCO(10),NPCO,IFPCO                            00000530
      COMMON/COMP1/ACO(10),ACW(10),ACG(10),PCO(10),PCW(10),PCG(10)         00000540
      COMMON/COMP2/NCO,NCW,NCG,IFCO,IFCW,IFCG                              00000550
      DIMENSION LDD(500),AK(500)                                           00000560
      EQUIVALENCE (AK(1),TKHY(1)),(BW,DELX)                                00000570
C  NTYPE = -1; CONSIDER GRAVITY EFFECTS                                    00000580
C  NTYPE =  1; CONSIDER CAPILLARY EFFECTS                                  00000590
C  NTYPE =  0; CONSIDER NEITHER                                            00000600
C  NTYPE =  2; CONSIDER BOTH                                               00000610
      READ(IN,1001) METHOD,NAP,NCELLS,NETA,NIC,IFRST,ILAST,NTYPE,NFIP,NL00000620
     &MDA,NSAT,NPOT,NCOMP,NTKH,NPMTX,NPLKUP                                00000630
      DT=-1.0                                                              00000640
1001  FORMAT(20I4)                                                         00000650
      NROW=NCELLS/NETA                                                     00000660
      READ(IN,1002) DELX,DELY,CF                                           00000670
1002  FORMAT(8F10.0)                                                       00000680
      IF(DELX.LE.0.0) GO TO 40                                             00000690
      DO 50 I=1,NETA                                                       00000700
50    DX(I)=DELX                                                           00000710
      DO 60 I=1,NROW                                                       00000720
60    DY(I)=DELY                                                           00000730
      GO TO 70                                                             00000740
40    READ(IN,1002) (DX(I),I=1,NETA)                                       00000750
      READ(IN,1002) (DY(I),I=1,NROW)                                       00000760
70    READ(IN,1001) (ICN(I),I=1,NIC)                                       00000770
      CALL WRTMOD                                                          00000780
      NF=(NCELLS+3)/4                                                      00000790
      READ(IN,1004) (ICT(I),I=1,NF)                                        00000800
1004  FORMAT(8I8)                                                          00000810
      CALL INPUT(D,NPMTX)                                                  00000820
      CALL INPUT(AK,NPMTX)                                                 00000830
      AKA=AK(IFRST)                                                        00000840
      CALL INPUT(HT,NPMTX)                                                 00000850
C  TRANSMISSILILITY CALCULATION                                            00000860
      NF=IFRST+1                                                           00000870
      KY=1+(IFRST-1)/NETA                                                  00000880
      KX=IFRST-NETA*(KY-1)                                                 00000890
      TKHX(IFRST)=AK(IFRST)*HT(IFRST)*DY(KY)/DX(KX)                        00000900
```

```
      TKHY(IFRST)=AK(IFRST)*HT(IFRST)*DX(KX)/DY(KY)                          00000910
      DO 61 I=NF,ILAST                                                       00000920
      IF(HT(I).EQ.0.0) GO TO 61                                              00000930
      AKA=AKA+AK(I)                                                          00000940
      KY=1+(I-1)/NETA                                                        00000950
      KX=I-NETA*(KY-1)                                                       00000960
      TKHX(I)=AK(I)*HT(I)*DY(KY)/DX(KX)                                      00000970
      TKHY(I)=AK(I)*HT(I)*DX(KX)/DY(KY)                                      00000980
      IF(KX.EQ.1.OR.HT(I-1).EQ.0.0) GO TO 64                                 00000990
      TKHX(I-1)=(TKHX(I)+TKHX(I-1))/(DX(KX)+DX(KX-1))                        00001000
64    IF(I.LE.NETA.OR.HT(I-NETA).EQ.0.0) GO TO 61                            00001010
      TKHY(I-NETA)=(TKHY(I)+TKHY(I-NETA))/(DY(KY)+DY(KY-1))                  00001020
61    CONTINUE                                                               00001030
      AKA=AKA/(ILAST-IFRST-NIC+1)                                            00001040
      IF(NTKH.EQ.0) GO TO 55                                                 00001050
      WRITE(IO,1008)                                                         00001060
1008  FORMAT('1TRANSMISSIBILITY MATRICES')                                   00001070
      DO 56 I=NETA,NCELLS,NETA                                               00001080
      WRITE(IO,1007) (TKHX(I+J-NETA),J=1,NETA)                               00001090
1007  FORMAT(//,' X-DIR.',2X,10E12.4)                                        00001100
      WRITE(IO,1011) (TKHY(I+J-NETA),J=1,NETA)                               00001110
1011  FORMAT(' Y-DIR.',2X,10E12.4)                                           00001120
56    CONTINUE                                                               00001130
55    IF(NTYPE.NE.-1.AND.NTYPE.NE.2) GO TO 53                                00001140
      DO 54 I=IFRST,ILAST                                                    00001150
C  SET UP GRAVITY * THICKNESS TERM AND CONVERT UNITS                         00001160
54    GH(I)=.2236111*HT(I)                                                   00001170
53    CALL INPUT(PHI,NPMTX)                                                  00001180
      CALL INPUT(SO,NPMTX)                                                   00001190
      CALL INPUT(SG,NPMTX)                                                   00001200
      CALL INPUT(SW,NPMTX)                                                   00001210
      CALL INPUT(ZEL,NPMTX)                                                  00001220
      CALL INPUT(LDD,NPMTX)                                                  00001230
      IF(NPLKUP.NE.0) WRITE(IO,1003)                                         00001240
1003  FORMAT('1')                                                            00001250
      CALL INDATA(AUW,PUW,NUW,IFUW)                                          00001260
      CALL INDATA(ABO,PBO,NBO,IFBO)                                          00001270
      CALL INDATA(ABW,PBW,NBW,IFBW)                                          00001280
      CALL INDATA(AKROG,SKROG,NKROG,IFKROG)                                  00001290
      CALL INDATA(AKROW,SKROW,NKROW,IFKROW)                                  00001300
      CALL INDATA(AKRG,SKRG,NKRG,IFKRG)                                      00001310
      CALL INDATA(AKRW,SKRW,NKRW,IFKRW)                                      00001320
      CALL INDATA(APCO,PPCO,NPCO,IFPCO)                                      00001330
      CALL INDATA(APCG,PPCG,NPCG,IFPCG)                                      00001340
      CALL INDATA(ACO,PCO,NCO,IFCC)                                          00001350
```

```
      CALL INDATA(ACW,PCW,NCW,IFCW)                                          00001360
      CALL INDATA(ACG,PCG,NCG,IFCG)                                          00001370
      READ(IN,1002) POWC,PGOC,ZOWC,ZGOC                                      00001380
      READ(IN,1002) (H(I),I=1,NAP)                                           00001390
C  THETA = 0. EXPLICIT;  THETA = 1. IMPLICIT                                 00001400
      READ(IN,1002) TF,THETA,EPS,T                                           00001410
      CALL ONCE                                                              00001420
      WRITE(IO,1006) CF,THETA,EPS,T,TF                                       00001430
1006  FORMAT('1ROCK COMP. FACTOR =',E13.6,//,' THETA =',F7.4,//,' CONVER00001440
     &GENCE TOLERANCE =',E10.3,//,' SYSTEM TEMPERATURE IS',F6.1,' DEGREE00001450
     &S FAHRENHEIT.',//,' TIME SPAN TO BE SIMULATED IS',F7.1,' DAYS.')   00001460
      J=1                                                                    00001470
      DO 90 I=IFRST,ILAST                                                    00001480
      IF(ICN(J).NE.I) GO TO 92                                               00001490
      J=J+1                                                                  00001500
      GO TO 90                                                               00001510
92    NF=1+(I-1)/NETA                                                        00001520
      VOL(I)=DX(I-NETA*(NF-1))*DY(NF)*HT(I)*PHI(I)                           00001530
      NF=N*(I-1)                                                             00001540
      CALL FLASH(D(I),NF,I)                                                  00001550
      CALL CRTPRP(D(I),NF,I)                                                 00001560
      CALL LOOKUP(PCWO(I),SO(I),APCO,PPCO,NPCO,IFPCO)                        00001570
      CALL LOOKUP(PCGO(I),SG(I),APCG,PPCG,NPCG,IFPCG)                        00001580
      CALL LOOKUP(CO,D(I),ACO,PCO,NCO,IFCO)                                  00001590
      CALL LOOKUP(CW,D(I),ACW,PCW,NCW,IFCW)                                  00001600
      CALL LOOKUP(CG,D(I),ACG,PCG,NCG,IFCG)                                  00001610
      B1(I)=CF+SO(I)*CO+SW(I)*CW+SG(I)*CG                                    00001620
90    CONTINUE                                                               00001630
      IF(D(IFRST).NE.0.0) GO TO 51                                           00001640
      DO 52 I=IFRST,ILAST                                                    00001650
C  FIND INITIAL PRESSURE FROM WATER-OIL CONTACT PRESSURE MINUS DENSITY*    00001660
C    GRAVITY*THICKNESS                                                      00001670
52    D(I)=PWOC-DL(I)*.2236111*ZEL(I)                                        00001680
      WRITE(IO,1009)                                                         00001690
1009  FORMAT('1ESTIMATED INITIAL PRESSURE BASED ON DENSITY, FORMATION EL00001700
     &EVATION, AND WATER-OIL CONTACT')                                   00001710
      DO 58 I=NETA,NCELLS,NETA                                               00001720
      WRITE(IO,1010) (D(I+J-NETA),J=1,NETA)                                  00001730
1010  FORMAT(//,10F10.0)                                                     00001740
58    CONTINUE                                                               00001750
51    IF(METHOD.EQ.2) GO TO 81                                               00001760
80    CALL IADIP(AKA,NTYPE)                                                  00001770
      STOP                                                                   00001780
81    CALL LSOR(NTYPE)                                                       00001790
      STOP                                                                   00001800
```

```
      END                                                                    00001810
C                                                                            00001820
C                                                                            00001830
      SUBROUTINE WRTMOD                                                      00001840
C  PURPOSE OF SUBROUTINE:                                                    00001850
C    TO WRITE A REPRESENTATION OF THE GRID STRUCTURE READ INTO PROGRAM        00001860
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW        00001870
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                  00001880
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP              00001890
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                         00001900
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                            00001910
      DATA NS1,NS2,NS4,NS2P,NS2B,NB1,NB2,NB4,NB2P,NP1/'/','//','////',      00001920
     &'//+',' //  ',' ',' ',  ',' ',  +',' +'/                                00001930
      WRITE(IO,1001) (DX(I),I=1,NETA)                                        00001940
1001  FORMAT('1',20X,'NUMBERING SCHEME OF RESERVOIR MODEL (IMAGINARY CEL     00001950
     &LS SHADED)',//,13X,'DX',4X,10(F7.1,2X))                                00001960
      M=9*NETA+1                                                             00001970
      DO 2 KK=1,4                                                            00001980
      WRITE(IO,1002) (NS1,I=1,M),(NS1,I=1,8)                                 00001990
1002  FORMAT(10X,8('/'),100A1)                                               00002000
2     CONTINUE                                                               00002010
      WRITE(IO,1003)                                                         00002020
1003  FORMAT('+     DY')                                                     00002030
      K=1                                                                    00002040
      DO 10 J=NETA,NCELLS,NETA                                               00002050
      DO 20 I=1,NETA                                                         00002060
      KK=J-NETA+I                                                            00002070
      IF(KK.LT.IFRST.OR.KK.GT.ILAST) GO TO 31                                00002080
      IF(ICN(K).EQ.KK) GO TO 32                                              00002090
      N=NB2                                                                  00002100
      NN=NB4                                                                 00002110
      NNN=NB2P                                                               00002120
      GO TO 30                                                               00002130
32    K=K+1                                                                  00002140
31    N=NS2                                                                  00002150
      NN=NS4                                                                 00002160
      NNN=NS2P                                                               00002170
30    NPQ(I)=N                                                               00002180
      NC(I)=KK                                                               00002190
      L=I+NETA                                                               00002200
      NPQ(L)=NN                                                              00002210
20    NPQ(L+NETA)=NNN                                                        00002220
      WRITE(IO,1002) (NP1,I=1,M),(NS1,I=1,8)                                 00002230
      WRITE(IO,1004) ((NPQ(I+(L-1)*NETA),L=1,3),I=1,NETA),NS2,NS4,NS2B       00002240
1004  FORMAT(10X,8('/'),'+',11(A2,A4,A3))                                    00002250
```

```
      WRITE(IO,1005) DY(J/NETA),(NPQ(I),NC(I),NPQ(I+2*NETA),I=1,NETA)   00002260
1005  FORMAT(3X,F7.1,8('/'),'+',11(A2,I3,1X,A3))                        00002270
      WRITE(IO,1006) (NB1,I=1,M),(NS1,I=1,8)                           00002280
1006  FORMAT('+',17X,100A1)                                            00002290
      WRITE(IO,1004) ((NPQ(I+(L-1)*NETA),L=1,3),I=1,NETA),NS2,NS4,NS2B 00002300
10    CONTINUE                                                         00002310
      WRITE(IO,1002) (NP1,I=1,M),(NS1,I=1,8)                           00002320
      DO 60 KK=1,4                                                     00002330
      WRITE(IO,1002) (NS1,I=1,M),(NS1,I=1,8)                           00002340
60    CONTINUE                                                         00002350
      RETURN                                                           00002360
      END                                                              00002370
C                                                                      00002380
C                                                                      00002390
      SUBROUTINE INPUT(E,NPMTX)                                        00002400
C  PURPOSE OF SUBROUTINE:                                              00002410
C    TO READ DATA MATRICES, ASSIGNING A VALUE TO EACH CELL OF SYSTEM   00002420
C      AND CHECK THAT IT LIES WITHIN THE RANGE SPECIFIED FOR THE DATA  00002430
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW  00002440
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST            00002450
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP        00002460
      DIMENSION E(NCELLS),NX(10)                                       00002470
      READ(IN,1001) (NX(I),I=1,10),FACTOR,XMIN,XMAX                    00002480
1001  FORMAT(10A4,4F10.0)                                              00002490
      IF(NPMTX.NE.0) WRITE(IO,1002) (NX(I),I=1,10),FACTOR,XMIN,XMAX    00002500
1002  FORMAT('1',5X,10A4,5X,'FACTOR = ',F9.4,5X,'MIN VALUE =',F9.4,5X, 00002510
     &'MAX VALUE =',F9.4,/)                                            00002520
      IF(IFRST.EQ.1) GO TO 61                                          00002530
      J1=IFRST-1                                                       00002540
      DO 64 I=1,J1                                                     00002550
64    E(I)=0.0                                                         00002560
61    IF(ILAST.EQ.NCELLS) GO TO 62                                     00002570
      J1=ILAST+1                                                       00002580
      DO 63 I=J1,NCELLS                                                00002590
63    E(I)=0.0                                                         00002600
62    IF(XMAX.LE.XMIN) GO TO 40                                        00002610
      DO 10 I=NETA,NCELLS,NETA                                         00002620
      READ(IN,1003) (E(J+I-NETA),J=1,NETA)                             00002630
1003  FORMAT(8F10.0)                                                   00002640
      IF(NPMTX.NE.0) WRITE(IO,1004) (E(J+I-NETA),J=1,NETA)             00002650
1004  FORMAT(/,11F11.4)                                                00002660
10    CONTINUE                                                         00002670
      J1=1                                                             00002680
      DO 20 I=IFRST,ILAST                                              00002690
      IF(E(I).GT.XMAX) GO TO 31                                        00002700
```

```
      IF(E(I).GE.XMIN) GO TO 21                                          00002710
      J2=J1                                                              00002720
      DO 30 J=J2,NIC                                                     00002730
      IF(I.NE.ICN(J)) GO TO 30                                           00002740
      J1=J                                                               00002750
      E(I)=0.0                                                           00002760
      GO TO 20                                                           00002770
30    CONTINUE                                                           00002780
31    WRITE(IO,1005) I                                                   00002790
1005  FORMAT(//,' CELL NUMBER',I4,' OF THE ABOVE MATRIX IS OUTSIDE THE B00002800
     &OUNDS SPECIFIED FOR THE DATA.',//,10X,'EXECUTION TERMINATING')     00002810
      NX(1)=-1                                                           00002820
21    E(I)=E(I)*FACTOR                                                   00002830
20    CONTINUE                                                           00002840
      IF(NX(1).LT.0) STOP                                                00002850
      RETURN                                                             00002860
40    DO 50 I=IFRST,ILAST                                                00002870
50    E(I)=XMIN                                                          00002880
      DO 51 I=1,NIC                                                      00002890
51    E(ICN(I))=0.0                                                      00002900
      DO 60 I=NETA,NCELLS,NETA                                           00002910
      IF(NPMTX.NE.0) WRITE(IO,1004) (E(J+I-NETA),J=1,NETA)               00002920
60    CONTINUE                                                           00002930
      RETURN                                                             00002940
      END                                                                00002950
C                                                                        00002960
C                                                                        00002970
      SUBROUTINE INDATA(A,P,KN,IFLAG)                                    00002980
C  PURPOSE OF SUBROUTINE:                                                00002990
C     TO READ IN DATA AS TABLE OF LOOKUP OR POLYNOMIAL COEFFICIENTS      00003000
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP          00003010
      DIMENSION A(10),P(10),NX(10)                                       00003020
      READ(IN,1001) (NX(I),I=1,10),KN,IFLAG                              00003030
1001  FORMAT(10A4,2I5)                                                   00003040
      IF(IFLAG.GT.1) GO TO 10                                            00003050
C  READ POLYNOMIAL COEFFICIENTS                                          00003060
      READ(IN,1002) (A(I),I=1,KN)                                        00003070
1002  FORMAT(6F13.0)                                                     00003080
      IF(NPLKUP.NE.0) WRITE(IO,1003) (NX(I),I=1,10),(A(I),I=1,KN)        00003090
1003  FORMAT(///,' POLYNOMIAL COEFFICIENTS FOR ',10A4,/,(/,E14.6))       00003100
      RETURN                                                             00003110
C  READ TABLE OF LOOK-UP INFORMATION                                     00003120
10    PI=0.0                                                             00003130
      DO 20 I=1,KN                                                       00003140
      READ(IN,1004) P(I),A(I)                                            00003150
```

```
1004  FORMAT(2F10.0)                                                          00003160
      IF(PI.GT.P(I)) IFLAG=10                                                 00003170
20    PI=P(I)                                                                 00003180
      IF(NPLKUP.NE.0) WRITE(IO,1005) (NX(I),I=1,10),(P(I),A(I),I=1,KN)        00003190
1005  FORMAT(///,' TABLE OF LOOK-UP INFORMATION FOR ',10A4,/,5X,'DEPENDE00003200
     &NT',5X,'INDEPENDENT',/,(/,2E16.6))                                      00003210
      IF(IFLAG.LE.9) RETURN                                                   00003220
      WRITE(IO,1006) (NX(I),I=1,10)                                           00003230
1006  FORMAT(//,' INDEPENDENT VARIABLES IN TABLE OF LOOK-UP FOR ',10A4,'00003240
     & ARE NOT IN ASCENDING ORDER.',/,10X,'EXECUTION TERMINATED.')            00003250
      STOP                                                                    00003260
      END                                                                     00003270
C                                                                             00003280
C                                                                             00003290
      SUBROUTINE SORSNK(AKQ,PWF,S,L,NIND)                                     00003300
C   PURPOSE OF SUBROUTINE:                                                    00003310
C     TO DETERMINE THE FLOW AT A SOURCE OR A SINK GIVEN THE FLOWING           00003320
C       BOTTOM HOLE PRESSURE                                                  00003330
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW         00003340
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                    00003350
      COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                       00003360
      COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                       00003370
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP               00003380
      COMMON/VSCTY/UMO(500),UMG(500),UMW(500)                                 00003390
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                             00003400
      CCMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                  00003410
      COMMON/RPERM/TKRO(500),TKRW(500),TKRG(500)                              00003420
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                            00003430
      COMMON/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)                           00003440
      COMMON/FVOLF/ABO(10),PBO(10),NBO,IFBO,ABW(10),PBW(10),NBW,IFBW          00003450
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                          00003460
      EQUIVALENCE (AVMG,BG),(TLAG,VB),(PIP1,DPP1),(PIM1,DPM1),(PIPN,DPPN00003470
     &),(PIMN,DPMN),(TLAO,TLAW)                                               00003480
      DATA RW/.666667/                                                        00003490
C   INDICATOR = 1 FOR INJECTION WELL                                          00003500
      I=NPQ(L)                                                                00003510
      CALL LOOKUP(BO,PO(I),ABO,PBO,NBO,IFBO)                                  00003520
      CALL LOOKUP(BW,PO(I),ABW,PBW,NBW,IFBW)                                  00003530
      K=N*(I-1)                                                               00003540
      IF(Y(K+1).EQ.0.0) GO TO 12                                              00003550
      AVMG=0.0                                                                00003560
      DO 40 M=1,N                                                             00003570
40    AVMG=AVMG+Y(M+K)*WM(M)                                                  00003580
      BG=379.306*DG(I)/AVMG                                                   00003590
12    M=1+(I-1)/NETA                                                          00003600
```

```
      K=I-NETA*(M-1)                                                    00003610
      IF(NIND.EQ.1.OR.PWF.EQ.0.0) GO TO 11                              00003620
      QO(L)=7.07*TKRO(I)*AKQ*HT(I)*(PO(I)-PWF)/(UMO(I)*BO*ALOG(SQR      00003630
     &T(DX(K)*DY(M)/3.141593)/RW)-.75+S)                                00003640
      CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PO)                              00003650
      DPP1=PIP1-PWF                                                     00003660
      DPM1=PIM1-PWF                                                     00003670
      DPPN=PIPN-PWF                                                     00003680
      DPMN=PIMN-PWF                                                     00003690
      DPT=DPP1+DPM1+DPPN+DPMN                                           00003700
      TLAO=(TLAMOX(I)*DPP1+SLAMOX*DPM1+TLAMOY(I)*DPPN+SLAMOY*DPMN)/DPT  00003710
      TLAG=(TLAMGX(I)*DPP1+SLAMGX*DPM1+TLAMGY(I)*DPPN+SLAMGY*DPMN)/DPT  00003720
      QG(L)=TLAG/TLAO*QO(L)                                             00003730
      TLAW=(TLAMWX(I)*DPP1+SLAMWX*DPM1+TLAMWY(I)*DPPN+SLAMWY*DPMN)/DPT  00003740
      QW(L)=TLAW/TLAG*QG(L)                                             00003750
      GO TO 10                                                          00003760
11    PWF=PO(I)-QO(L)*UMO(I)*BO*(ALOG(SQRT(DX(K)*DY(M)/3.141593)/RW)-.75 00003770
     &+S)/(7.07*TKRO(I)*AKQ*HT(I))                                      00003780
10    VB=DX(K)*DY(M)*HT(I)/886.905                                      00003790
      WRITE(IO,1001) I,PWF,QO(L),QG(L),QW(L),NIND                       00003800
1001  FORMAT(I4,4E18.6,I4)                                              00003810
      IF(QO(L).EQ.0.0) GO TO 20                                         00003820
      QO(L)=QO(L)*UMO(I)*BO/VB                                          00003830
20    IF(QG(L).EQ.0.0) GO TO 30                                         00003840
      QG(L)=QG(L)*UMG(I)*BG/VB                                          00003850
30    IF(QW(L).EQ.0.0) RETURN                                           00003860
      QW(L)=QW(L)*UMW(I)*BW/VB                                          00003870
      RETURN                                                            00003880
      END                                                               00003890
C                                                                       00003900
C                                                                       00003910
      SUBROUTINE LOOKUP(B,PI,A,P,KN,IFLAG)                              00003920
C  PURPOSE OF SUBROUTINE:                                               00003930
C    TO DETERMINE THE VALUE OF THE DATA REQUIRED BY USING A TABLE       00003940
C      OF LOOKUP OR CALCULATING THE VALUE OF THE POLYNOMIAL AS REQUIRED 00003950
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP         00003960
      DIMENSION A(10),P(10)                                             00003970
      IF(PI.EQ.0.0) RETURN                                              00003980
      IF(IFLAG.GT.1) GO TO 10                                           00003990
C  POLYNOMIAL CALCULATION                                               00004000
      B=0.0                                                             00004010
      DO 11 I=1,KN                                                      00004020
11    B=B*PI+A(KN+1-I)                                                  00004030
      RETURN                                                            00004040
C  FIND RANGE                                                           00004050
```

```
10      J=1                                                                    00004060
        IF(PI.LT.P(1)) GO TO 21                                                00004070
        DO 20 I=2,KN                                                           00004080
        IF(PI-P(I)) 30,50,20                                                   00004090
20      J=I                                                                    00004100
21      WRITE(IO,1001) PI                                                      00004110
1001    FORMAT('1P VALUE OF',E14.6,' OUT OF RANGE',//,' EXECUTION STOPS')      00004120
        STOP                                                                   00004130
30      IF(IFLAG.GT.2) GO TO 40                                                00004140
C   LINEAR INTERPOLATION                                                       00004150
        B=A(J)+(PI-P(J))/(P(J+1)-P(J))*(A(J+1)-A(J))                           00004160
        RETURN                                                                 00004170
C   LA GRANGIAN INTERPOLATION                                                  00004180
40      B=(PI-P(J+1))/(P(J)-P(J+1))*(PI-P(J+2))/(P(J)-P(J+2))*A(J)+(PI-P(J     00004190
       &))/(P(J+1)-P(J+2))*((PI-P(J+2))/(P(J+1)-P(J))*A(J+1)-(PI-P(J+1))/(     00004200
       &P(J+2)-P(J))*A(J+2))                                                   00004210
        RETURN                                                                 00004220
50      B=A(J+1)                                                               00004230
        RETURN                                                                 00004240
        END                                                                    00004250
C                                                                              00004260
C                                                                              00004270
        SUBROUTINE ACCP                                                        00004280
C   PURPOSE OF SUBROUTINE:                                                     00004290
C     TO DETERMINE THE ACCELERATION PARAMETER VALUES TO BE USED                00004300
C       IN THE IADIP METHOD WHEN NONE ARE SPECIFIED                            00004310
        COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                        00004320
        COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW        00004330
        COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                 00004340
        COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                  00004350
        COMMON/TKH/TKHX(500),TKHY(500)                                         00004360
        EQUIVALENCE (TY,AM4),(AM3,R)                                           00004370
        DATA TX,TY/2*0.0/                                                      00004380
        DO 60 I=IFRST,ILAST                                                    00004390
        IF(HT(I).EQ.0.0) GO TO 60                                              00004400
        TX=TX+TKHX(I)/HT(I)                                                    00004410
        TY=TY+TKHY(I)/HT(I)                                                    00004420
60      CONTINUE                                                               00004430
        TX=TX/(ILAST-IFRST-NIC+1)                                              00004440
        TY=TY/(ILAST-IFRST-NIC+1)                                              00004450
        AM2=2.*TX/(TX+TY)                                                      00004460
        AM4=2.*TY/(TX+TY)                                                      00004470
        AM1=2.467401*AM2/NETA**2                                               00004480
        AM3=2.467401*AM4/NROW**2                                               00004490
        IF(AM3.LT.AM1) AM1=AM3                                                 00004500
```

```
      IF(AM4.GT.AM2) AM2=AM4                                                00004510
      R=EXP(ALOG(AM2/AM1)/(NAP-1))                                          00004520
      DO 30 I=2,NAP                                                         00004530
30    H(I)=H(I-1)*R                                                         00004540
      RETURN                                                                00004550
      END                                                                   00004560
C                                                                           00004570
C                                                                           00004580
      SUBROUTINE IADIP(AKA,NTYPE)                                           00004590
C  PURPOSE OF SUBROUTINE:                                                   00004600
C    TO PERFORM THE ITERATIVE OR NON-ITERATIVE ALTERNATING                  00004610
C      DIRECTION IMPLICIT PROCEDURE ON THE GIVEN GRID SYSTEM                00004620
      COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                       00004630
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                           00004640
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW       00004650
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                 00004660
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                          00004670
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                00004680
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP             00004690
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                              00004700
C  PO = LAST TIME STEP                                                      00004710
C  PN = NEW ITERATION(HALF TIME STEP)                                       00004720
C  PNP= LAST ITERATION(FULL TIME STEP)                                      00004730
      DATA M1,M2,M3,M4,TIME/'NON-',    ',' (AD',',(IAD',0.0/                00004740
      IF(H(1).LT.0.0) CALL ACCP                                             00004750
      AKA=.22*AKA                                                           00004760
      IF(NAP.EQ.1) GO TO 72                                                 00004770
      M1=M2                                                                 00004780
      M3=M4                                                                 00004790
72    WRITE(IO,1001) M1,M3                                                  00004800
1001  FORMAT(///,' METHOD OF SOLUTION:  ',A4,'ITERATIVE ALTERNATING DIRE    00004810
     &CTION PROCEDURE ',A2,'ADIP).')                                        00004820
      IF(NAP.NE.1) WRITE(IO,1006) (H(K),K=1,NAP)                            00004830
1006  FORMAT(///,6X,'H(K)',/,20(/,E14.6))                                   00004840
71    NCYCLE=0                                                              00004850
      DO 1 I=IFRST,ILAST                                                    00004860
1     PO(I)=D(I)                                                            00004870
      CALL IBM0B1                                                           00004880
      CALL OLDT(NTYPE)                                                      00004890
      TIME=TIME+DT                                                          00004900
      IF(NETA.NE.NCELLS) DT=DT/2.                                           00004910
70    NCYCLE=NCYCLE+1                                                       00004920
      IF(NCYCLE.GE.20) RETURN                                               00004930
      DO 800 K=1,NAP                                                        00004940
      DO 22 I=IFRST,ILAST                                                   00004950
```

```
22      PO(I)=D(I)                                                        00004960
C   X-DIRECTION SWEEP                                                     00004970
        DO 5 I=IFRST,ILAST                                                00004980
        KY=1+(I-1)/NETA                                                   00004990
        KX=I-NETA*(KY-1)                                                  00005000
        HKG=AKA*H(K)/(DX(KX)*DY(KY))                                      00005010
        F=THETA/DX(KX)                                                    00005020
        CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PNP)                             00005030
        CALL CPGRV(HTX,HTY,SLAMX,SLAMY,TLAMX,TLAMY,I,NTYPE)               00005040
        A(I)=F*SLAMX                                                      00005050
        C(I)=F*TLAMX                                                      00005060
        B(I)=-A(I)-C(I)-VOL(I)*B1(I)/DT-HKG                               00005070
5       D(I)=-THETA/DY(KY)*(TLAMY*(PIPN-PO(I))-SLAMY*(PO(I)-PIMN)+        00005080
       &HTY)-F*HTX+OT(I)-HKG*PO(I)                                        00005090
        DO 7 I=1,NIC                                                      00005100
7       B(ICN(I))=0.0                                                     00005110
        CALL TRIHOR(IFRST,ILAST)                                          00005120
        IF(NETA.EQ.NCELLS) GO TO 800                                      00005130
        DO 40 I=IFRST,ILAST                                               00005140
40      PN(I)=D(I)                                                        00005150
C   Y-DIRECTION SWEEP                                                     00005160
        DO 6 I=IFRST,ILAST                                                00005170
        KY=1+(I-1)/NETA                                                   00005180
        KX=I-NETA*(KY-1)                                                  00005190
        HKG=AKA*H(K)/(DX(KX)*DY(KY))                                      00005200
        F=THETA/DY(KY)                                                    00005210
        CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PN)                              00005220
        CALL CPGRV(HTX,HTY,SLAMX,SLAMY,TLAMX,TLAMY,I,NTYPE)               00005230
        A(I)=F*SLAMY                                                      00005240
        C(I)=F*TLAMY                                                      00005250
        B(I)=-A(I)-C(I)-VOL(I)*B1(I)/DT-HKG                               00005260
6       D(I)=-THETA/DX(KX)*(TLAMX*(PIP1-PN(I))-SLAMX*(PN(I)-PIMN)+HT      00005270
       &X)-F*HTY+OT(I)-HKG*PN(I)                                          00005280
        DO 8 I=1,NIC                                                      00005290
8       B(ICN(I))=0.0                                                     00005300
        CALL TRIVRT                                                       00005310
800     CONTINUE                                                          00005320
        DO 50 I=IFRST,ILAST                                               00005330
50      PNP(I)=D(I)                                                       00005340
        CALL UPDATE                                                       00005350
        IF(NAP.EQ.1) GO TO 73                                             00005360
        NC=0                                                              00005370
C   CHECK FOR CONVERGENCE                                                 00005380
        DO 60 I=IFRST,ILAST                                               00005390
60      IF(ABS(PO(I)-D(I)).LE.ABS(EPS*D(I))) NC=NC+1                      00005400
```

```
      IF(NC.LT.ILAST-IFRST+1) GO TO 70                                  00005410
73    WRITE(IO,1003) TIME,M3                                            00005420
1003  FORMAT('1AT',F8.2,' DAYS, THE ',A4,'IP) PRESSURE DISTRIBUTION IS')00005430
      DO 2 J=NETA,NCELLS,NETA                                           00005440
      WRITE(IO,1004) (PNP(J+I-NETA),I=1,NETA)                           00005450
1004  FORMAT(//,10F10.0)                                                00005460
2     CONTINUE                                                          00005470
      IF(NAP.NE.1) WRITE(IO,1005) NCYCLE                                00005480
1005  FORMAT('-CONVERGENCE OBTAINED IN ',I2,' CYCLES.')                 00005490
      IF(TIME.GE.TF) RETURN                                             00005500
      GO TO 71                                                          00005510
      END                                                               00005520
C                                                                       00005530
C                                                                       00005540
      SUBROUTINE LSOR(NTYPE)                                            00005550
C  PURPOSE OF SUBROUTINE:                                               00005560
C    TO PERFORM THE LINE SUCCESSIVE OVER-RELAXATION PROCEDURE           00005570
C      ON THE GIVEN GRID SYSTEM                                         00005580
      COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                   00005590
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                       00005600
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW   00005610
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST             00005620
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)            00005630
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                      00005640
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP         00005650
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                          00005660
      EQUIVALENCE (H(1),OMEGA)                                          00005670
      TIME=0.0                                                          00005680
      WRITE(IO,1001) OMEGA                                              00005690
1001  FORMAT(///,' METHOD OF SOLUTION:  LINE SUCCESSIVE OVER-RELAXATION 00005700
     &PROCEDURE (LSOR)',//,' OMEGA =',F7.4)                             00005710
      DO 9 I=1,NCELLS                                                   00005720
9     PN(I)=D(I)                                                        00005730
71    DO 8 I=IFRST,ILAST                                                00005740
8     PO(I)=PNP(I)                                                      00005750
      CALL IBM0B1                                                       00005760
      CALL OLDT(NTYPE)                                                  00005770
      TIME=TIME+DT                                                      00005780
      ITER=0                                                            00005790
30    NC=0                                                              00005800
      ITER=ITER+1                                                       00005810
      IF(ITER.GT.20) RETURN                                             00005820
C  SWEEP OF EACH ROW                                                    00005830
      DO 20 J=NETA,NCELLS,NETA                                          00005840
      JMNETA=J-NETA                                                     00005850
```

```
      DO 39 K=1,NETA                                                    00005860
      I=JMNETA+K                                                        00005870
      CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PNP)                             00005880
      CALL CPGRV(HTX,HTY,SLAMX,SLAMY,TLAMX,TLAMY,I,NTYPE)               00005890
      KY=1+(I-1)/NETA                                                   00005900
      F=THETA/DX(I-NETA*(KY-1))                                         00005910
      A(K)=F*SLAMX                                                      00005920
      C(K)=F*TLAMX                                                      00005930
      B(K)=-A(K)-C(K)-F*(TLAMY+SLAMY)-VOL(I)*B1(I)/DT                   00005940
39    D(K)=-THETA/DY(KY)*(TLAMY*PIPN+SLAMY*PIMN+HTY)-F*HTX+OT(I)        00005950
      NFIRST=0                                                          00005960
      DO 38 I=1,NETA                                                    00005970
      IF(B(I).EQ.0.0) GO TO 38                                          00005980
      IF(NFIRST.EQ.0) NFIRST=I                                          00005990
      NLAST=I                                                           00006000
38    CONTINUE                                                          00006010
      CALL TRIHOR(NFIRST,NLAST)                                         00006020
      DO 20 K=1,NETA                                                    00006030
      I=JMNETA+K                                                        00006040
C  CHECK FOR CONVERGENCE                                                00006050
      IF(ABS(D(K)-PNP(I)).LE.ABS(EPS*PNP(I))) NC=NC+1                   00006060
20    PNP(I)=PNP(I)+OMEGA*(D(K)-PNP(I))                                 00006070
      CALL UPDATE                                                       00006080
      IF(NC.LT.NCELLS) GO TO 30                                         00006090
      WRITE(IO,1002) TIME                                               00006100
1002  FORMAT('1AT',F7.1,' DAYS, THE (LSOR) PRESSURE DISTRIBUTION IS')   00006110
      DO 2 J=NETA,NCELLS,NETA                                           00006120
      WRITE(IO,1003) (PNP(J+I-NETA),I=1,NETA)                           00006130
1003  FORMAT(//,10F10.0)                                                00006140
2     CONTINUE                                                          00006150
      WRITE(IO,1004) ITER                                               00006160
1004  FORMAT('-CONVERGENCE OBTAINED IN',I3,' ITERATIONS.')              00006170
      IF(TIME.GE.TF) RETURN                                             00006180
      GO TO 71                                                          00006190
      END                                                               00006200
C                                                                       00006210
C                                                                       00006220
      SUBROUTINE BOUND(I,PIP1,PIM1,PIPN,PIMN,P)                         00006230
C  PURPOSE OF SUBROUTINE:                                               00006240
C    TO DETERMINE THE VALUES TO BE USED FOR SURROUNDING CELLS           00006250
C      BASED ON THEIR POSITION IN THE MODEL                             00006260
      COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                   00006270
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW   00006280
      DIMENSION P(NCELLS)                                               00006290
      M=(I-1)/4+1                                                       00006300
```

```
      K=I-(M-1)*4                                                          00006310
      K=100**(5-K)                                                         00006320
      L=ICT(M)/K                                                           00006330
      L=ICT(M)-L*K                                                         00006340
      L=100*L/K                                                            00006350
      ENTRY EBOUND(PIP1,PIM1,PIPN,PIMN,P)                                  00006360
      PIP1=P(I)                                                            00006370
      PIM1=P(I)                                                            00006380
      PIPN=P(I)                                                            00006390
      PIMN=P(I)                                                            00006400
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16),L                     00006410
3     PIP1=P(I+1)                                                          00006420
      GO TO 9                                                              00006430
5     PIP1=P(I+1)                                                          00006440
4     PIM1=P(I-1)                                                          00006450
9     PIPN=P(I+NETA)                                                       00006460
      RETURN                                                               00006470
7     PIM1=P(I-1)                                                          00006480
1     PIMN=P(I-NETA)                                                       00006490
12    PIP1=P(I+1)                                                          00006500
      RETURN                                                               00006510
8     PIPN=P(I+NETA)                                                       00006520
2     PIM1=P(I-1)                                                          00006530
      GO TO 11                                                             00006540
14    PIP1=P(I+1)                                                          00006550
10    PIM1=P(I-1)                                                          00006560
      RETURN                                                               00006570
15    PIM1=P(I-1)                                                          00006580
6     PIP1=P(I+1)                                                          00006590
13    PIPN=P(I+NETA)                                                       00006600
11    PIMN=P(I-NETA)                                                       00006610
16    RETURN                                                               00006620
      END                                                                  00006630
C                                                                          00006640
C                                                                          00006650
      SUBROUTINE TRIHOR(NFIRST,NLAST)                                      00006660
C  PURPOSE OF SUBROUTINE:                                                  00006670
C    TO PERFORM THE HORIZONAL VERSION OF THE THOMAS ALGORITHM FOR          00006680
C      A TRIDIAGONAL SYSTEM                                                00006690
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW      00006700
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                00006710
      C(NFIRST)=C(NFIRST)/B(NFIRST)                                        00006720
      D(NFIRST)=D(NFIRST)/B(NFIRST)                                        00006730
      N1=NFIRST+1                                                          00006740
      DO 10 I=N1,NLAST                                                     00006750
```

```
      IF(B(I).EQ.0.0) GO TO 10                                          00006760
      DENOM=B(I)-A(I)*C(I-1)                                            00006770
      C(I)=C(I)/DENOM                                                   00006780
      D(I)=(D(I)-A(I)*D(I-1))/DENOM                                     00006790
10    CONTINUE                                                          00006800
      N1=NLAST-1                                                        00006810
      DO 20 I=NFIRST,N1                                                 00006820
      J=N1-I+NFIRST                                                     00006830
      IF(B(I).EQ.0.0) GO TO 20                                          00006840
      D(J)=D(J)-C(J)*D(J+1)                                             00006850
20    CONTINUE                                                          00006860
      RETURN                                                            00006870
      END                                                               00006880
C                                                                       00006890
C                                                                       00006900
      SUBROUTINE TRIVRT                                                 00006910
C  PURPOSE OF SUBROUTINE:                                               00006920
C    TO PERFORM THE VERTICAL VERSION OF THE THOMAS ALGORITHM FOR        00006930
C      A TRIDIAGONAL SYSTEM                                             00006940
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW   00006950
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST             00006960
      DO 20 I=IFRST,ILAST                                               00006970
      IF(B(I).EQ.0.0) GO TO 21                                          00006980
      IF(I.LE.NETA) GO TO 30                                            00006990
      IMNETA=I-NETA                                                     00007000
      DENOM=B(I)-A(I)*A(IMNETA)                                         00007010
      D(I)=(D(I)-A(I)*D(IMNETA))/DENOM                                  00007020
      IF(I.LE.NCELLS-NETA) A(I)=C(I)/DENOM                              00007030
      GO TO 20                                                          00007040
30    A(I)=C(I)/B(I)                                                    00007050
      D(I)=D(I)/B(I)                                                    00007060
      GO TO 20                                                          00007070
21    A(I)=0.0                                                          00007080
20    CONTINUE                                                          00007090
      L=NCELLS-NETA                                                     00007100
      DO 70 I=1,L                                                       00007110
      N=L-I+1                                                           00007120
      IF(B(I).EQ.0.0) GO TO 70                                          00007130
      D(N)=D(N)-A(N)*D(NCELLS+1-I)                                      00007140
70    CONTINUE                                                          00007150
      RETURN                                                            00007160
      END                                                               00007170
C                                                                       00007180
C                                                                       00007190
      SUBROUTINE ONCE                                                   00007200
```

```
C  PURPOSE OF SUBROUTINE:                                                    00007210
C    TO READ IN ALL OF THE FLUID PROPERTY DATA REQUIRED FOR THE            00007220
C      GIVEN SYSTEM AND TO PERFORM ALL OF THE ONE TIME CALCULATIONS        00007230
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW      00007240
      COMMON/CRIT/PC(10),TC(10),TR(10),VC(10),ZC(10)/KVAL/CC(10),PRS(10)  00007250
      COMMON/PCNV/A(10),B(10)/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)      00007260
      COMMON/VLIQ/DLQ(10),TA,TB,XK(10),XM(10),NC7,NCR                     00007270
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                      00007280
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP           00007290
      COMMON/TRIDAG/A1(500),B1(500),C1(500),D(500),IFRST,ILAST            00007300
      DIMENSION W(10),Z(5000)                                             00007310
      EQUIVALENCE (W(1),DLQ(1)),(C,ZETA,ZT),(TA,XN,PC1),(TB,XM1,ZI),(XK1  00007320
     &,TC1),(X(1),Z(1))                                                   00007330
C IF C7+ FRACTION NOT PRESENT, NC7=N+1                                     00007340
      READ(IN,1001) NCR,NC7,N                                             00007350
1001  FORMAT(3I4)                                                         00007360
      DO 82 I=1,N                                                         00007370
      CALL INPUT(Y,0)                                                     00007380
      DO 82 J=IFRST,ILAST                                                 00007390
82    Z(N*(J-1)+I)=Y(J)                                                   00007400
      IF(NCR.EQ.N) GO TO 20                                               00007410
      READ(IN,1002)((NC(I+(J-1)*5),I=1,5),PC(J),TC(J),VC(J),ZC(J),WM(J),  00007420
     &A(J),B(J),W(J),J=1,NCR)                                             00007430
1002  FORMAT(5A2,5F10.0,/,10X,3F10.0)                                     00007440
      J=1                                                                 00007450
      NL=1                                                                00007460
80    READ(IN,1001) NFRST,NLAST                                           00007470
C  FOR LAST GROUPED COMPONENT, NFRST=NCR+1; NLAST=-1                       00007480
C  FOR PURE BETWEEN GROUPS, NFRST = FIRST OF NEXT GROUP; NLAST = LAST OF  00007490
C        NEXT GROUP                                                       00007500
      IF(NFRST.EQ.NL) GO TO 85                                            00007510
      NF=NFRST-1                                                          00007520
      DO 84 I=NL,NF                                                       00007530
      NA=(J-1)*5                                                          00007540
      NB=(I-1)*5                                                          00007550
      DO 83 L=1,5                                                         00007560
83    NC(L+NA)=NC(L+NB)                                                   00007570
      PC(J)=PC(I)                                                         00007580
      TC(J)=TC(I)                                                         00007590
      VC(J)=VC(I)                                                         00007600
      ZC(J)=ZC(I)                                                         00007610
      WM(J)=WM(I)                                                         00007620
      DO 88 L=IFRST,ILAST                                                 00007630
88    Z(N*(L-1)+J)=Z(N*(L-1)+I)                                           00007640
      A(J)=A(I)                                                           00007650
```

```
      B(J)=B(I)                                                          00007660
      W(J)=(.291-ZC(J))/.08                                              00007670
84    J=J+1                                                              00007680
85    IF(NLAST.LE.0) GO TO 50                                            00007690
      READ(IN,1002) (NC(I+(J-1)*5),I=1,5)                                00007700
      ZT=0.0                                                             00007710
      PC1=0.0                                                            00007720
      TC1=0.0                                                            00007730
      VC1=0.0                                                            00007740
      ZC1=0.0                                                            00007750
      WM1=0.0                                                            00007760
C   SET UP FOR ALL Z'S ORIGINALLY SAME FOR EACH CELL                     00007770
      DO 81 I=NFRST,NLAST                                                00007780
      ZI=Z(N*(IFRST-1)+I)                                                00007790
      ZT=ZT+ZI                                                           00007800
      PC1=PC1+PC(I)*ZI                                                   00007810
      TC1=TC1+TC(I)*ZI                                                   00007820
      VC1=VC1+VC(I)*ZI                                                   00007830
      ZC1=ZC1+ZC(I)*ZI                                                   00007840
81    WM1=WM1+WM(I)*ZI                                                   00007850
      PC(J)=PC1/ZT                                                       00007860
      TC(J)=TC1/ZT                                                       00007870
      VC(J)=VC1/ZT                                                       00007880
      ZC(J)=ZC1/ZT                                                       00007890
      WM(J)=WM1/ZT                                                       00007900
      DO 89 L=IFRST,ILAST                                                00007910
89    Z(N*(L-1)+J)=ZT                                                    00007920
      A(J)=9400.*WM(J)**(-1.71)                                          00007930
      B(J)=2.*WM(J)**(-.115)                                             00007940
86    W(J)=(.291-ZC(J))/.08                                              00007950
      NL=NLAST+1                                                         00007960
      J=J+1                                                              00007970
      GO TO 80                                                           00007980
20    DO 10 J=1,N                                                        00007990
      READ(IN,1003) (NC(I+(J-1)*5),I=1,5),PC(J),TC(J),VC(J),ZC(J),WM(J),00008000
     &DLQ(J),A(J),B(J),XK1,XN,XM1,C                                      00008010
1003  FORMAT(5A2,6F10.0,/,10X,6F10.0)                                    00008020
      IF(J.GE.NC7) GO TO 10                                              00008030
C   CALCULATE ALANI-KENNEDY PURE COMPONENT VALUES                        00008040
      XK(J)=XK1*EXP(XN/T)                                                00008050
      XM(J)=XM1*T+C                                                      00008060
10    CONTINUE                                                           00008070
50    DO 90 J=1,N                                                        00008080
      XK1=T/TC(J)                                                        00008090
      TR(J)=XK1                                                          00008100
```

```
      XM1=ZC(J)                                                          00008110
      SRWM(J)=SQRT(WM(J))                                                00008120
C  CALCULATE ZETA FACTOR (REID & SHERWOOD, P. 405)                       00008130
      ZETA=5.44085*TC(J)**.166667/SRWM(J)/PC(J)**.666667                 00008140
C  CALCULATE STIEL-THODOS LOW PRESSURE, PURE GAS CORRELATION (R.&S.,P405 00008150
      IF(XK1.GT.1.5) GO TO 30                                            00008160
      UMU(J)=34.E-5*XK1**.94/ZETA                                        00008170
      GO TO 40                                                           00008180
30    UMU(J)=17.78E-5*(4.58*XK1-1.67)**.625/ZETA                         00008190
C  CALCULATE CONSTANTS FOR EQUILK                                        00008200
40    CC(J)=XM1*(.132+.25/XK1)                                           00008210
90    PRS(J)=10.**((((90.*XM1-73.85)*XM1+16.25)*(XK1-1.)/XK1)-(10.**(-8. 00008220
     &68*(XK1-1.8+6.2*XM1)**2)))                                         00008230
C  CALCULATE TEMPERATURE RELATED FUNCTIONS (CONSTANT)                    00008240
      TA=261.8082/T+10.75352                                             00008250
      TB=2.232395E-4*T+6.225655                                          00008260
      WRITE(IO,1004)                                                     00008270
1004  FORMAT('1',8X,'NAME',10X,'PC',10X,'TC',10X,'VC',10X,'ZC',10X,'WM', 00008280
     &11X,'A',11X,'B',9X,'DL OR W',/)                                    00008290
      DO 60 J=1,N                                                        00008300
      NA=(J-1)*5                                                         00008310
      WRITE(IO,1005) (NC(I+NA),I=1,5),PC(J),TC(J),VC(J),ZC(J),WM(J),A(J) 00008320
     &,B(J),W(J)                                                         00008330
1005  FORMAT(5X,5A2,2(5X,F7.2),2(5X,F7.4),5X,F7.3,5X,F8.4,5X,F7.4,6X,F7. 00008340
     &4,/)                                                               00008350
60    CONTINUE                                                           00008360
      IF(NCOMP.EQ.0) RETURN                                              00008370
      WRITE(IO,1006)                                                     00008380
1006  FORMAT('1OVERALL COMPOSITION MATRIX')                              00008390
      DO 51 I=NETA,NCELLS,NETA                                           00008400
      WRITE(IO,1007) (NC(J),J=1,5),(Z(N*(I+J-NETA)-9),J=1,NETA)          00008410
1007  FORMAT(//,1X,5A2,10F10.6)                                          00008420
      DO 51 K=2,N                                                        00008430
      NA=(K-1)*5                                                         00008440
      WRITE(IO,1008) (NC(J+NA),J=1,5),(Z(N*(I+J-NETA-1)+K),J=1,NETA)     00008450
1008  FORMAT(1X,5A2,10F10.6)                                             00008460
51    CONTINUE                                                           00008470
      RETURN                                                             00008480
      END                                                                00008490
C                                                                        00008500
C                                                                        00008510
      FUNCTION CUBEQN(P,Q,R)                                             00008520
C  PURPOSE OF SUBPROGRAM:                                                00008530
C    TO DETERMINE THE SOLUTION TO A CUBIC EQUATION GIVEN THE             00008540
C      COEFFICIENTS                                                      00008550
```

```
      EQUIVALENCE (SA,CUBEQN)                                          00008560
      P=P/3.                                                           00008570
C CALCULATE SMALL A FOR CUBIC EQUATION ROOT                            00008580
      SA=Q/3.-P*P                                                      00008590
C CALCULATE SMALL B FOR CUBIC EQUATION ROOT                            00008600
      SB=P**3-.5*(P*Q-R)                                               00008610
C CALCULATE SQUARE ROOT FACTOR                                         00008620
      F=SB*SB+SA**3                                                    00008630
      IF(F.LE.0.0) GO TO 10                                            00008640
      F=SQRT(F)                                                        00008650
C DETERMINE ROOT OF CUBIC EQUATION                                     00008660
      IF(F+SB.LT.0.0) GO TO 30                                         00008670
      IF(F-SB.LT.0.0) GO TO 50                                         00008680
      CUBEQN=((-SB+F)**.333333)-((SB+F)**.333333)-P                    00008690
      RETURN                                                           00008700
50    CUBEQN=(-(SB-F)**.333333)-((SB+F)**.333333)-P                    00008710
      RETURN                                                           00008720
30    IF(F-SB.LT.0.0) GO TO 60                                         00008730
      CUBEQN=((-SB+F)**.333333)+((-SB-F)**.333333)-P                   00008740
      RETURN                                                           00008750
60    CUBEQN=(-(SB-F)**.333333)+((-SB-F)**.333333)-P                   00008760
      RETURN                                                           00008770
C  USE TRIG METHOD TO FIND ROOT                                        00008780
10    F=2.*SQRT(-SA)                                                   00008790
      CUBEQN=F*COS(((ARCOS(2.*SB/(SA*F)))+6.283185)/3.)-P              00008800
      RETURN                                                           00008810
      END                                                              00008820
C                                                                      00008830
C                                                                      00008840
      SUBROUTINE FLASH(P,L,M)                                          00008850
C  PURPOSE OF SUBROUTINE:                                              00008860
C    TO PERFORM A FLASH CALCULATION ON A GIVEN FLUID SYSTEM            00008870
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                   00008880
      DIMENSION RK(10),Z(5000)                                         00008890
      EQUIVALENCE (X(1),Z(1))                                          00008900
      CALL EQUILK(P,RK,L)                                              00008910
      IF(XL(M).GT.0.05.AND.XL(M).LT.0.95) GO TO 51                     00008920
C  CALCULATE FUNCTION FOR LIQUID PHASE = 0.0 AND 1.0                   00008930
70    F0=0.0                                                           00008940
      F1=0.0                                                           00008950
      DO 10 I=1,N                                                      00008960
      XLN=Z(L+I)*(1.-RK(I))                                            00008970
      F0=F0+XLN/RK(I)                                                  00008980
      FH=XLN/(.5*(1.+RK(I)))                                           00008990
10    F1=F1+XLN                                                        00009000
```

```
      IF(F0.LT.0.0) GO TO 22                                                    00009010
      IF(F1.GT.0.0) GO TO 32                                                    00009020
C  INITIAL GUESS USING LAGRANGIAN INTERPOLATION                                 00009030
      XL(M)=F0/(FH-F1)*(-.5*F1/(F0-FH)+FH/(F0-F1))                              00009040
51    IT=0                                                                      00009050
C  USE NEWTON-RAPHSON METHOD FOR CONVERGENCE                                    00009060
50    IT=IT+1                                                                   00009070
      IF(IT.GT.50) GO TO 60                                                     00009080
      XN=0.0                                                                    00009090
      XD=0.0                                                                    00009100
      DO 40 I=1,N                                                               00009110
      F1=Z(L+I)*(1.-RK(I))/(XLN*(1.-RK(I))+RK(I))                               00009120
      XN=XN+F1                                                                  00009130
40    XD=XD+F1*F1/Z(L+I)                                                        00009140
      XL(M)=XLN+XN/XD                                                           00009150
      IF(XL(M).LT.0.0) XL(M)=0.0                                                00009160
      IF(XL(M).GT.1.0) XL(M)=1.0                                                00009170
      IF(ABS(XL(M)-XLN).LE.(.001*XLN)) GO TO 60                                 00009180
      XLN=XL(M)                                                                 00009190
      GO TO 50                                                                  00009200
22    XL(M)=0.0                                                                 00009210
20    DO 21 I=1,N                                                               00009220
      K=L+I                                                                     00009230
      Y(K)=Z(K)                                                                 00009240
21    X(K)=0.0                                                                  00009250
      RETURN                                                                    00009260
32    XL(M)=1.0                                                                 00009270
30    DO 31 I=1,N                                                               00009280
31    Y(L+I)=0.0                                                                00009290
      RETURN                                                                    00009300
60    IF(XL(M).EQ.0.0) GO TO 20                                                 00009310
      IF(XL(M).EQ.1.0) GO TO 30                                                 00009320
      DO 61 I=1,N                                                               00009330
      K=L+I                                                                     00009340
      X(K)=Z(K)/(XL(M)+(1.-XL(M))*RK(I))                                        00009350
61    Y(K)=RK(I)*X(K)                                                           00009360
      RETURN                                                                    00009370
      END                                                                       00009380
C                                                                               00009390
C                                                                               00009400
      SUBROUTINE EQUILK(P,RK,L)                                                 00009410
C  PURPOSE OF SUBROUTINE:                                                       00009420
C    TO DETERMINE THE EQUILIBRIUM K-VALUES FOR A GIVEN SYSTEM                   00009430
      COMMON/CRIT/PC(10),TC(10),TR(10),VC(10),ZC(10)/KVAL/CC(10),PRS(10)00009440
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                            00009450
```

```
      DIMENSION RK(N)                                                    00009460
C  DETERMINE CONVERGENCE PRESSURE OF GIVEN SYSTEM                        00009470
      CALL PCONV(PC,PCON,L)                                              00009480
      IF(PCON.LT.1.1*P) PCON=1.1*P                                       00009490
      POP=1.0-(P/PCON)                                                   00009500
      EXPF=.00014*PCON+.08                                               00009510
      DO 10 I=1,N                                                        00009520
      PR=P/PC(I)                                                         00009530
C  DETERMINE EQUILIBRIUM K-VALUES                                        00009540
10    RK(I)=((PRS(I)/PR)*EXP(CC(I)*(PR-PRS(I))))**(POP**(EXPF/TR(I)))    00009550
C  THE NEXT 10 ARE TEMPORARY TO CORRECT INACCURACIES IN THE CURRENT      00009560
C    K-VALUE ALGORITHM*************************************************00009570
      RK(1)=2.0                                                          00009580
      RK(2)=1.075                                                        00009590
      RK(3)=0.88                                                         00009600
      RK(4)=0.64                                                         00009610
      RK(5)=0.58                                                         00009620
      RK(6)=0.40                                                         00009630
      RK(7)=0.25                                                         00009640
      RK(8)=0.16                                                         00009650
      RK(9)=0.0                                                          00009660
      RK(10)=0.0                                                         00009670
      RETURN                                                             00009680
      END                                                                00009690
C                                                                        00009700
C                                                                        00009710
      SUBROUTINE PCONV(PC,PCON,L)                                        00009720
C  PURPOSE OF SUBROUTINE:                                                00009730
C    TO DETERMINE THE CONVERGENCE PRESSURE FOR A GIVEN SYSTEM            00009740
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                     00009750
      COMMON/PCNV/A(10),B(10)/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)     00009760
      COMMON/VLIQ/DLQ(10),TA,TB,XK(10),XM(10),NC7,NCR                    00009770
      DIMENSION PC(N),W(10),Z(5000)                                      00009780
      EQUIVALENCE (Z(1),X(1))                                            00009790
      AV=0.0                                                             00009800
      PCON=6000.                                                         00009810
      IF(N.NE.NC7) PCON=PC(N)                                            00009820
      DO 10 I=1,N                                                        00009830
      W(I)=Z(L+I)*WM(I)                                                  00009840
10    AV=AV+W(I)                                                         00009850
      DO 20 I=1,N                                                        00009860
C  CALCULATE WEIGHT FRACTION OF EACH COMPONENT                           00009870
20    W(I)=W(I)/AV                                                       00009880
      E1=Z(L+1)                                                          00009890
      IF(WM(1).LE.17.) E1=W(1)                                           00009900
```

```
      PCON=A(1)*(E1**B(1))*(AV-WM(1))+PC(1)*Z(L+1)+PCON*Z(L+N)                00009910
      E1=1.                                                                  00009920
      E3=1.                                                                  00009930
      NN=N-1                                                                 00009940
      I=1                                                                    00009950
      DO 30 J=2,NN                                                           00009960
      E1=E1-Z(L+I)                                                           00009970
      AV=AV-WM(I)*W(I)                                                       00009980
      E3=E3-W(I)                                                             00009990
      I=J                                                                    00010000
C  CALCULATE CONVERGENCE PRESSURE                                           00010010
30    PCON=PCON+(A(J)*((Z(L+J)/E1)**B(J))*(AV/E3-WM(J))*E1)+PC(J)*Z(L+J)     00010020
      RETURN                                                                 00010030
      END                                                                    00010040
C                                                                            00010050
C                                                                            00010060
      SUBROUTINE CRTPRP(P,L,M)                                               00010070
C  PURPOSE OF SUBROUTINE:                                                   00010080
C    TO DETERMINE THE CRITICAL PROPERTIES FOR A GIVEN SYSTEM                 00010090
      COMMON/CRIT/PC(10),TC(10),TR(10),VC(10),ZC(10)/CRITM/PCMG,PCML,TCM     00010100
     &G,TCML,VCMG,VCML,ZCMG,ZCML/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)      00010110
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                         00010120
      EQUIVALENCE (XI,YI,PCMG)                                               00010130
      AVMG=0.0                                                               00010140
      TCMG=0.0                                                               00010150
      ZCMG=0.0                                                               00010160
      VCMG=0.0                                                               00010170
      AVML=0.0                                                               00010180
      TCML=0.0                                                               00010190
      ZCML=0.0                                                               00010200
      VCML=0.0                                                               00010210
      DO 10 I=1,N                                                            00010220
C  CALCULATE LIQUID MIXTURE CRITICAL PROPERTIES                             00010230
      XI=X(L+I)                                                              00010240
      AVML=AVML+WM(I)*XI                                                     00010250
      TCML=TCML+TC(I)*XI                                                     00010260
      ZCML=ZCML+ZC(I)*XI                                                     00010270
      VCML=VCML+VC(I)*XI                                                     00010280
C  CALCULATE GAS MIXTURE CRITICAL PROPERTIES                                00010290
      YI=Y(L+I)                                                              00010300
      AVMG=AVMG+WM(I)*YI                                                     00010310
      TCMG=TCMG+TC(I)*YI                                                     00010320
      ZCMG=ZCMG+ZC(I)*YI                                                     00010330
10    VCMG=VCMG+VC(I)*YI                                                     00010340
      IF(Y(L+1).EQ.0.0) GO TO 20                                             00010350
```

```
      PCMG=ZCMG*10.73*TCMG/VCMG                                          00010360
      IF(X(L+N).EQ.0.0) GO TO 30                                         00010370
20    PCML=ZCML*10.73*TCML/VCML                                          00010380
30    CALL VOLLIQ(P,L,M)                                                 00010390
      CALL VOLGAS(P,L,M)                                                 00010400
      CALL VISCTY(P,L,M)                                                 00010410
      RETURN                                                             00010420
      END                                                                00010430
C                                                                        00010440
C                                                                        00010450
      SUBROUTINE VOLLIQ(P,L,M)                                           00010460
C  PURPOSE OF SUBROUTINE:                                                00010470
C    TO DETERMINE THE LIQUID VOLUME FOR A GIVEN SYSTEM                   00010480
      COMMON/CRIT/PC(10),TC(10),TR(10),VC(10),ZC(10)                     00010490
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                     00010500
      COMMON/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)                      00010510
      COMMON/VLIQ/DLQ(10),TA,TB,XK(10),XM(10),NC7,NCR                    00010520
      COMMON/DNSTY/DL(500),DW(500),DG(500)                               00010530
      DIMENSION W(10)                                                    00010540
      EQUIVALENCE (W(1),DLQ(1)),(BM,XC7),(XI,A1,S),(Q,B),(R,A3,C),(WMC7,00010550
     &C1),(DC7,C,C2),(A,A4)                                              00010560
      DL(M)=0.0                                                          00010570
      IF(X(L+1).EQ.0.0) RETURN                                           00010580
      IF(NCR.NE.N) GO TO 40                                              00010590
      AM=0.0                                                             00010600
      BM=0.0                                                             00010610
      IF(NC7.GT.N) GO TO 20                                              00010620
      DC7=0.0                                                            00010630
      WMC7=0.0                                                           00010640
      DO 30 I=NC7,N                                                      00010650
      XI=X(L+I)                                                          00010660
      XC7=XC7+XI                                                         00010670
      DC7=DC7+DLQ(I)*XI                                                  00010680
30    WMC7=WMC7+WM(I)*XI                                                 00010690
C  DETERMINE C7+ FRACTION MOLECULAR WEIGHT AND DENSITY                   00010700
      WMC7=WMC7/XC7                                                      00010710
      DC7=DC7/XC7                                                        00010720
C  CALCULATE ALANI-KENNEDY C7+ FRACTION A AND B VALUES                   00010730
      AM=XC7*EXP((7.310446E-6*WMC7-5.967829E-2/DC7+3.840599E-3)*WMC7+TA)00010740
      BM=XC7*((3.499274E-2-1.018528/DC7)*WMC7-1.165471E-1*DC7+TB)        00010750
20    NN=NC7-1                                                           00010760
      DO 10 I=1,NN                                                       00010770
      XI=X(L+I)                                                          00010780
      AM=AM+XK(I)*XI                                                     00010790
10    BM=BM+XM(I)*XI                                                     00010800
```

```
      A1=-BM-10.73*T/P                                                    00010810
      AM=AM/P                                                             00010820
      BM=-BM*AM                                                           00010830
      DL(M)=AVML/CUBEQN(A1,AM,BM)                                         00010840
      RETURN                                                              00010850
40    C1=0.0                                                              00010860
      C2=0.0                                                              00010870
      B=0.0                                                               00010880
      A1=0.0                                                              00010890
      A2=0.0                                                              00010900
      A3=0.0                                                              00010910
      A4=0.0                                                              00010920
      DO 50 I=1,N                                                         00010930
      J=L+I                                                               00010940
      AM=115.133*TC(I)/PC(I)                                              00010950
      A1=A1+X(J)*(AM*TC(I)*(.25913-.031314*W(I)))**0.5                    00010960
      A2=A2+X(J)*(AM*(.0249+.15369*W(I)))**0.5                            00010970
      TC2=TC(I)**2                                                        00010980
      BM=AM*TC2                                                           00010990
      A3=A3+X(J)*(BM*(.2015+.21642*W(I)))**0.5                            00011000
      A4=A4+X(J)*TC2*(BM*.042*W(I))**0.5                                  00011010
      B=B+X(J)*AM*.009152                                                 00011020
      BM=BM*.64277/PC(I)                                                  00011030
      C1=C1+X(J)*(BM*TC(I)**.5*(1.-W(I)))**.333333                        00011040
50    C2=C2+X(J)*(BM*TC2*(.302584+1.53486*W(I)))**.333333                 00011050
      A=A1*A1-A2*A2*T+(A4*A4/T**4+A3*A3)/T                                00011060
      C=C1**3/T**.5+C2**3/T**2                                            00011070
      R=(A*B-C)/P                                                         00011080
      S=-10.73*T/P                                                        00011090
      Q=A/P+(S-B)*B                                                       00011100
      DL(M)=AVML/CUBEQN(S,Q,R)                                            00011110
      RETURN                                                              00011120
      END                                                                 00011130
C                                                                         00011140
C                                                                         00011150
      SUBROUTINE VOLGAS(P,L,M)                                            00011160
C   PURPOSE OF SUBROUTINE:                                                00011170
C     TO DETERMINE THE VAPOR VOLUME FOR A GIVEN SYSTEM                    00011180
      COMMON/CRITM/PCMG,PCML,TCMG,TCML,VCMG,VCML,ZCMG,ZCML                00011190
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                      00011200
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                00011210
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP           00011220
      COMMON/TRIDAG/A1(500),B1(500),C(500),D(500),IFRST,ILAST             00011230
      COMMON/FVOLF/ABO(10),PBO(10),NBO,IFBO,ABW(10),PBW(10),NBW,IFBW      00011240
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)              00011250
```

```
      COMMON/SATN/SW(500),SG(500),SO(500),CF                                    00011260
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                               00011270
      COMMON/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)                             00011280
      DIMENSION A(36),B(12)                                                     00011290
      EQUIVALENCE (XX,YY,BM),(X2,Y2,ZZ),(BO,BG,BW)                              00011300
      DATA A/2.14335,.331235,.105729,-.052184,.019704,-5.30959E-3,.08317000011310
     &62,-.134036,-.0503937,.0443121,-.0263833,8.91783E-3,-.021467,.066800011320
     &81,5.09248E-3,-.0193295,.0192621,-.0108948,-8.71403E-4,-.0271743,.00011330
     &015513,5.89725E-3,-.0115354,9.55939E-3,4.28463E-3,8.85123E-3,-7.310011340
     &819E-3,1.53667E-3,4.29101E-3,-6.0114E-3,-1.65953E-3,-2.15209E-3,2.00011350
     &696E-3,-2.83268E-3,-8.13025E-4,3.11752E-3/                                00011360
      DATA B(1),B(7),GIP,OIP,WIP/2*7.071068E-1,3*0.0/                           00011370
      DG(M)=0.0                                                                 00011380
      IF(HT(M).EQ.0.0) RETURN                                                   00011390
      CALL LOOKUP(BW,P,ABW,PBW,NBW,IFBW)                                        00011400
      DW(M)=63.698/BW                                                           00011410
      IF(M.NE.IFRST) GO TO 11                                                   00011420
      GIP=0.0                                                                   00011430
      OIP=0.0                                                                   00011440
      WIP=0.0                                                                   00011450
      VTOT=0.0                                                                  00011460
11    IF(Y(L+1).EQ.0.0) GO TO 20                                                00011470
      XX=1.351351E-1*P/PCMG-1.013514                                            00011480
      X2=XX*XX                                                                  00011490
      B(2)=1.224745*XX                                                          00011500
      B(3)=2.371709*X2-7.905695E-1                                              00011510
      B(4)=(4.677073*X2-2.806244)*XX                                            00011520
      B(5)=(9.280775*X2-7.95495)*X2+.795495                                     00011530
      B(6)=((1.846851E1*X2-2.052057E1)*X2+4.397265)*XX                          00011540
      YY=1.052632*T/TCMG-2.105263                                               00011550
      Y2=YY*YY                                                                  00011560
      B(8)=1.224745*YY                                                          00011570
      B(9)=2.371709*Y2-7.905695E-1                                              00011580
      B(10)=(4.677073*Y2-2.806244)*YY                                           00011590
      B(11)=(9.280775*Y2-7.95495)*Y2+.795495                                    00011600
      B(12)=((1.846851E1*Y2-2.052057E1)*Y2+4.397265)*YY                         00011610
      ZZ=0.0                                                                    00011620
      DO 10 K=1,6                                                               00011630
      BM=B(K)                                                                   00011640
      J=6*K-6                                                                   00011650
      DO 10 LL=1,6                                                              00011660
10    ZZ=ZZ+A(J+LL)*BM*B(LL+6)                                                  00011670
      DG(M)=AVMG*P/(ZZ*10.73*T)                                                 00011680
20    IF(NFIP.EQ.0) RETURN                                                      00011690
C  CALCULATE AND PRINT FLUIDS IN PLACE                                          00011700
```

```
      VTOT=VTOT+VOL(M)                                                          00011710
      WIP=WIP+VOL(M)*SW(M)/(5.6146*BW)                                          00011720
      IF(X(L+1).EQ.0.0) GO TO 21                                                00011730
      CALL LOOKUP(BO,P,ABO,PBO,NBO,IFBO)                                        00011740
      OIP=OIP+VOL(M)*SO(M)/(5.6146*BO)                                          00011750
      IF(Y(L+1).EQ.0.0) GO TO 22                                                00011760
21    BG=379.306*DG(M)/AVMG                                                     00011770
      GIP=GIP+VOL(M)*SG(M)/BG                                                   00011780
22    IF(M.NE.ILAST) RETURN                                                     00011790
      WRITE(IO,1001) VTOT,GIP,OIP,WIP                                           00011800
1001  FORMAT('1TOTAL RESERVOIR PORE VOLUME =',E14.6,' CUBIC FEET',//,' G00011810
     &AS IN PLACE =',E14.6,' STANDARD CUBIC FEET',//,' OIP IN PLACE =',E00011820
     &14.6,' STOCK TANK BARRELS',//,' WATER IN PLACE =',E14.6,' STOCK TA00011830
     &NK BARRELS',/,3E18.6)                                                     00011840
      RETURN                                                                    00011850
      END                                                                       00011860
C                                                                               00011870
C                                                                               00011880
      SUBROUTINE VISCTY(P,L,M)                                                  00011890
C  PURPOSE OF SUBROUTINE:                                                       00011900
C    TO DETERMINE THE LIQUID AND VAPOR VISCOSITIES FOR A GIVEN SYSTEM           00011910
      COMMON/CRITM/PCMG,PCML,TCMG,TCML,VCMG,VCML,ZCMG,ZCML                      00011920
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                            00011930
      COMMON/VISC/AVMG,AVML,SRWM(10),UMU(10),WM(10)                             00011940
      COMMON/VSCTY/UMO(500),UMG(500),UMW(500)                                   00011950
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                      00011960
      COMMON/WTRV/AUW(10),PUW(10),NUW,IFUW                                      00011970
      EQUIVALENCE (DRMG,XD),(DRML,XDP),(XN,UMM)                                 00011980
      CALL LOOKUP(UMW(M),P,AUW,PUW,NUW,IFUW)                                    00011990
      IF(Y(L+1).NE.0.0) GO TO 40                                                00012000
      UMM=0.0                                                                   00012010
      GO TO 30                                                                  00012020
40    XDP=0.0                                                                   00012030
      XN=0.0                                                                    00012040
      DO 10 I=1,N                                                               00012050
      XD=SRWM(I)*Y(L+I)                                                         00012060
      XN=XN+XD*UMU(I)                                                           00012070
10    XDP=XDP+XD                                                                00012080
C  CALCULATE HERNING-ZIPPERER LOW PRESSURE GAS MIXTURE VISCOSITY (REID &00012090
C       SHERWOOD, P. 423)                                                       00012100
      UMM=XN/XDP                                                                00012110
C  CALCULATE REDUCED DENSITIES                                                  00012120
      UMO(M)=0.0                                                                00012130
      IF(X(L+N).EQ.0.0) GO TO 20                                                00012140
30    DRML=VCML/(AVML/DL(M))                                                    00012150
```

```
C   CALCULATE LIQUID VISCOSITY AT RESERVOIR CONDITIONS USING JOSSI-STIEL-00012160
C         THODOS CORRELATION (REID & SHERWOOD, P. 413)                   00012170
        UMO(M)=(((((.0093324*DRML-.040758)*DRML+.058533)*DRML+.023364)*DRM00012180
       &L+.1023)**4-.0001)/5.44085/TCML**.166667*SQRT(AVML)*PCML**.666667+00012190
       &UMM                                                             00012200
        UMG(M)=0.0                                                      00012210
        IF(Y(L+1).EQ.0.0) RETURN                                        00012220
20      DRMG=VCMG/(AVMG/DG(M))                                          00012230
C   CALCULATE GAS VISCOSITY AT RESERVOIR CONDITIONS USING DEAN-STIEL    00012240
C         CORRELATION (REID & SHERWOOD, P. 426)                         00012250
        UMG(M)=UMM+1.98498E-5*(EXP(1.439*DRMG)-EXP(-1.111*DRMG**1.858))/TC00012260
       &MG**.166667*SQRT(AVMG)*PCMG**.666667                            00012270
        RETURN                                                          00012280
        END                                                             00012290
C                                                                       00012300
C                                                                       00012310
        SUBROUTINE SATRN                                                00012320
C   PURPOSE OF SUBROUTINE:                                              00012330
C     TO DETERMINE THE OIL, GAS, AND WATER SATURATICNS FOR A GIVEN SYSTEM00012340
        COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW 00012350
        COMMON/SATN/SW(500),SG(500),SO(500),CF                          00012360
        COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST           00012370
        COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                    00012380
        COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)          00012390
        COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                 00012400
        COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)               00012410
        COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)               00012420
        COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                     00012430
        COMMON/DNSTY/DL(500),DW(500),DG(500)                            00012440
        COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP       00012450
        DATA N1,N2,N3/'OIL ','GAS ','H2O '/                             00012460
        J=1                                                             00012470
        KQ=1                                                            00012480
        DO 10 I=IFRST,ILAST                                             00012490
        IF(ICN(J).NE.I) GO TO 20                                        00012500
        J=J+1                                                           00012510
        GO TO 10                                                        00012520
20      PHIOLD=PHI(I)                                                   00012530
        PHI(I)=PHIOLD*(1.+CF*(PNP(I)-PO(I)))                            00012540
        VOL(I)=PHIOLD*VOL(I)/PHI(I)                                     00012550
        CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PNP)                          00012560
        KY=I+(I-1)/NETA                                                 00012570
        BQO=0.0                                                         00012580
        BQW=0.0                                                         00012590
        PFAC=(PHI(I)-PHICLD)*DT/PHICLD                                  00012600
```

```
      IF(NPQ(KQ).NE.I) GO TO 12                                          00012610
      IF(DL(I).NE.0.0) BQO=QO(KQ)*DT/(VOL(I)*DL(I))                      00012620
      IF(DW(I).NE.0.0) BQW=QW(KQ)*DT/(VOL(I)*DW(I))                      00012630
      KQ=KQ+1                                                            00012640
12    C2=DT/VOL(I)/DY(KY)                                                00012650
      C1=DT/VOL(I)/DX(I-NETA*(KY-1))                                     00012660
      IF(I.LE.NETA) GO TO 41                                             00012670
      SLAMOY=TLAMOY(I-NETA)                                              00012680
      SLAMWY=TLAMWY(I-NETA)                                              00012690
42    SLAMOX=TLAMOX(I-1)                                                 00012700
      SLAMWX=TLAMWX(I-1)                                                 00012710
      GO TO 45                                                           00012720
41    SLAMOY=0.0                                                         00012730
      SLAMWY=0.0                                                         00012740
      IF(I.NE.IFRST) GO TO 42                                            00012750
      SLAMOX=0.0                                                         00012760
      SLAMWX=0.0                                                         00012770
45    SO(I)=C1*(TLAMOX(I)*(PIP1-PNP(I))-SLAMOX*(PNP(I)-PIM1))+C2*(TLAMOY00012780
     &(I)*(PIPN-PNP(I))-SLAMOY*(PNP(I)-PIMN))-BQO-SO(I)*PFAC+SO(I)       00012790
      SW(I)=C1*(TLAMWX(I)*(PIP1-PNP(I))-SLAMWX*(PNP(I)-PIM1))+C2*(TLAMWY00012800
     &(I)*(PIPN-PNP(I))-SLAMWY*(PNP(I)-PIMN))-BQW-SW(I)*PFAC+SW(I)       00012810
      IF(SO(I).LT.1.E-5) SO(I)=0.0                                       00012820
      IF(SW(I).LT.1.E-5) SW(I)=0.0                                       00012830
      SG(I)=1.0-SW(I)-SO(I)                                              00012840
      IF(SG(I).LT.1.E-5) SG(I)=0.                                        00012850
10    CONTINUE                                                           00012860
      IF(NSAT.EQ.0) RETURN                                               00012870
      WRITE(IO,1001)                                                     00012880
1001  FORMAT('1SATURATION MATRICES')                                     00012890
      DO 51 I=NETA,NCELLS,NETA                                           00012900
      WRITE(IO,1002) N1,(SO(I+J-NETA),J=1,NETA)                          00012910
1002  FORMAT(//,1X,A4,10F10.5)                                           00012920
      WRITE(IO,1003) N2,(SG(I+J-NETA),J=1,NETA)                          00012930
1003  FORMAT(1X,A4,10F10.5)                                              00012940
      WRITE(IO,1003) N3,(SW(I+J-NETA),J=1,NETA)                          00012950
51    CONTINUE                                                           00012960
      RETURN                                                             00012970
      END                                                                00012980
C                                                                        00012990
C                                                                        00013000
      SUBROUTINE IBMOB1                                                  00013010
C  PURPOSE OF SUBROUTINE:                                                00013020
C    TO DETERMINE THE INTERBLOCK MOBILITIES FOR A GIVEN SYSTEM           00013030
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW    00013040
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)             00013050
```

```
      COMMON/SATN/SW(500),SG(500),SO(500),CF                                00013060
      COMMON/FLUID3/AKROG(10),AKROW(10),AKRG(10),AKRW(10)                   00013070
      COMMON/FLUID4/SKROG(10),SKROW(10),SKRG(10),SKRW(10)                   00013080
      COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                     00013090
      COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                     00013100
      COMMON/TKH/TKHX(500),TKHY(500)                                        00013110
      COMMON/PERMF/NKROG,IFKROG,NKROW,IFKROW,NKRG,IFKRG,NKRW,IFKRW          00013120
      COMMON/RPERM/TKRO(500),TKRW(500),TKRG(500)                            00013130
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                 00013140
      COMMON/VSCTY/UMO(500),UMG(500),UMW(500)                               00013150
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                          00013160
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP             00013170
      DATA NX,NY,N1,N2,N3/'X-DI','Y-DI','OIL ','GAS ','H2O '/              00013180
      J=1                                                                   00013190
C  CALCULATE KRO FOR EACH CELL USING STONE EQUATION                         00013200
      DO 10 I=IFRST,ILAST                                                   00013210
      IF(ICN(J).NE.I) GO TO 15                                              00013220
      J=J+1                                                                 00013230
      GO TO 10                                                              00013240
15    CALL LOOKUP(TKROW,SW(I),AKROW,SKROW,NKROW,IFKROW)                     00013250
      CALL LOOKUP(TKRW(I),SW(I),AKRW,SKRW,NKRW,IFKRW)                       00013260
      CALL LOOKUP(TKROG,SG(I),AKROG,SKROG,NKROG,IFKROG)                     00013270
      CALL LOOKUP(TKRG(I),SG(I),AKRG,SKRG,NKRG,IFKRG)                       00013280
      TKRO(I)=(TKROW+TKRW(I))*(TKROG+TKRG(I))-(TKRW(I)+TKRG(I))             00013290
      IF(TKRO(I).LT.0.0) TKRO(I)=0.0                                        00013300
10    CONTINUE                                                              00013310
C  CALCULATE INTERBLOCK MOBILITIES FOR EACH CELL                            00013320
      J=1                                                                   00013330
      LL=NCELLS-NETA+1                                                      00013340
      DO 30 I=IFRST,ILAST                                                   00013350
      IF(ICN(J).NE.I) GO TO 20                                              00013360
      J=J+1                                                                 00013370
      GO TO 30                                                              00013380
20    KY=1+(I-1)/NETA                                                       00013390
      L=KY*NETA                                                             00013400
      KX=I-NETA*(KY-1)                                                      00013410
      M=L-NETA+1                                                            00013420
      IF(L.GT.ILAST) L=ILAST                                                00013430
      IF(M.LT.IFRST) M=IFRST                                                00013440
      CALL IBMOB2(DX,TOX,TGX,TWX,I,KX,L,M,1)                                00013450
      CALL IBMOB2(DY,TOY,TGY,TWY,I,KX,LL,KY,NETA)                           00013460
      IF(UMO(I).EQ.0.0) GO TO 11                                            00013470
      TLAMOX(I)=TOX/UMO(I)                                                  00013480
      TLAMOY(I)=TOY/UMO(I)                                                  00013490
      GO TO 12                                                              00013500
```

```
11    TLAMOX(I)=0.0                                                    00013510
      TLAMOY(I)=0.0                                                    00013520
12    IF(UMG(I).EQ.0.0) GO TO 13                                       00013530
      TLAMGX(I)=TGX/UMG(I)                                             00013540
      TLAMGY(I)=TGY/UMG(I)                                             00013550
      GO TO 14                                                         00013560
13    TLAMGX(I)=0.0                                                    00013570
      TLAMGY(I)=0.0                                                    00013580
14    TLAMWX(I)=TWX/UMW(I)                                             00013590
      TLAMWY(I)=TWY/UMW(I)                                             00013600
      IF(I.EQ.L) GO TO 41                                              00013610
      IF(HT(I+1).EQ.0.0) GO TO 41                                      00013620
      IF(I.EQ.1) GO TO 50                                              00013630
      IF(PNP(I-1).GT.PNP(I)) GO TO 61                                  00013640
50    IF(I.GE.LL) GO TO 51                                             00013650
      IF(HT(I+NETA).EQ.0.0) GO TO 51                                   00013660
      IF(I.LE.NETA) GO TO 31                                           00013670
      IF(PNP(I-NETA).GT.PNP(I)) GO TO 71                               00013680
      GO TO 31                                                         00013690
41    TLAMOX(I)=0.0                                                    00013700
      TLAMGX(I)=0.0                                                    00013710
      TLAMWX(I)=0.0                                                    00013720
      GO TO 50                                                         00013730
51    TLAMOY(I)=0.0                                                    00013740
      TLAMGY(I)=0.0                                                    00013750
      TLAMWY(I)=0.0                                                    00013760
      GO TO 31                                                         00013770
61    TLAMOX(I)=TLAMOX(I-1)                                            00013780
      TLAMGX(I)=TLAMGX(I-1)                                            00013790
      TLAMWX(I)=TLAMWX(I-1)                                            00013800
      GO TO 50                                                         00013810
71    TLAMOY(I)=TLAMOY(I-NETA)                                         00013820
      TLAMGY(I)=TLAMGY(I-NETA)                                         00013830
      TLAMWY(I)=TLAMWY(I-NETA)                                         00013840
31    TLAMOX(I)=TLAMOX(I)*TKHX(I)                                      00013850
      TLAMGX(I)=TLAMGX(I)*TKHX(I)                                      00013860
      TLAMWX(I)=TLAMWX(I)*TKHX(I)                                      00013870
      TLAMOY(I)=TLAMOY(I)*TKHY(I)                                      00013880
      TLAMGY(I)=TLAMGY(I)*TKHY(I)                                      00013890
      TLAMWY(I)=TLAMWY(I)*TKHY(I)                                      00013900
30    CONTINUE                                                         00013910
      IF(NLMDA.EQ.0) RETURN                                            00013920
      WRITE(IO,1001) NX                                                00013930
1001  FORMAT('1CROSS SECTIONAL AREA * ',A4,'RECTION MOBILITIES')       00013940
      DO 53 L=NETA,NCELLS,NETA                                         00013950
```

```
      WRITE(IO,1002) N1,(TLAMOX(L+J-NETA),J=1,NETA)                    00013960
1002  FORMAT(//,1X,A4,10E12.4)                                         00013970
      WRITE(IO,1003) N2,(TLAMGX(L+J-NETA),J=1,NETA)                    00013980
1003  FORMAT(1X,A4,10E12.4)                                            00013990
      WRITE(IO,1003) N3,(TLAMWX(L+J-NETA),J=1,NETA)                    00014000
53    CONTINUE                                                         00014010
      IF(NETA.EQ.NCELLS) RETURN                                        00014020
      WRITE(IO,1001) NY                                                00014030
      DO 56 L=NETA,NCELLS,NETA                                         00014040
      WRITE(IO,1002) N1,(TLAMOY(L+J-NETA),J=1,NETA)                    00014050
      WRITE(IO,1003) N2,(TLAMGY(L+J-NETA),J=1,NETA)                    00014060
      WRITE(IO,1003) N3,(TLAMWY(L+J-NETA),J=1,NETA)                    00014070
56    CONTINUE                                                         00014080
      RETURN                                                           00014090
      END                                                              00014100
C                                                                      00014110
C                                                                      00014120
      SUBROUTINE IBMOB2(D,TO,TG,TW,I,KX,L,M,N)                         00014130
C  PURPOSE OF SUBROUTINE:                                              00014140
C    TO SUPPLEMENT IBMOB1 IN DETERMINING THE INTERBLOCK MOBILITIES     00014150
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                      00014160
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)           00014170
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                     00014180
      COMMON/RPERM/TKRO(500),TKRW(500),TKRG(500)                       00014190
      DIMENSION D(20)                                                  00014200
      N2=N*2                                                           00014210
10    IF(I.LT.L) GO TO 70                                              00014220
      IF(PNP(I).LT.PNP(I-N)) GO TO 30                                  00014230
20    TO=TKRO(I)                                                       00014240
      TW=TKRW(I)                                                       00014250
      TG=TKRG(I)                                                       00014260
      RETURN                                                           00014270
30    DO 31 J=1,NQ                                                     00014280
      IF(I-N.EQ.NPQ(J)) GO TO 20                                       00014290
31    CONTINUE                                                         00014300
50    IF(I-N2.LE.0) GO TO 40                                           00014310
      IF(HT(I-N).EQ.0.0.OR.HT(I-N2).EQ.0.0) GO TO 40                   00014320
51    DO 53 J=2,NQ                                                     00014330
      IF(I-N2.EQ.NPQ(J)) GO TO 40                                      00014340
53    CONTINUE                                                         00014350
60    F=D(KX-1)/(D(KX-1)+D(KX-2))                                      00014360
      TO=TKRO(I-N)+F*(TKRO(I-N)-TKRO(I-N2))                            00014370
      TW=TKRW(I-N)+F*(TKRW(I-N)-TKRW(I-N2))                            00014380
      TG=TKRG(I-N)+F*(TKRG(I-N)-TKRG(I-N2))                            00014390
61    IF(TO.LT.0.0) TO=0.0                                             00014400
```

```
      IF(TW.LT.0.0) TW=0.0                                                  00014410
      IF(TG.LT.0.0) TG=0.0                                                  00014420
      RETURN                                                                00014430
70    IF(PNP(I).GT.PNP(I+N)) GO TO 80                                       00014440
      DO 71 J=1,NQ                                                          00014450
      IF(I.EQ.NPQ(J).OR.I+N.EQ.NPQ(J)) GO TO 20                             00014460
71    CONTINUE                                                              00014470
      IF(HT(I+N).EQ.0.0) GO TO 20                                           00014480
      F=D(KX)/(D(KX)+D(KX+1))                                               00014490
      TO=TKRO(I)-F*(TKRO(I+N)-TKRO(I))                                      00014500
      TW=TKRW(I)-F*(TKRW(I+N)-TKRW(I))                                      00014510
      TG=TKRG(I)-F*(TKRG(I+N)-TKRG(I))                                      00014520
      GO TO 61                                                              00014530
80    IF(I.EQ.M) GO TO 20                                                   00014540
      IF(PNP(I).GE.PNP(I-N)) GO TO 20                                       00014550
      DO 81 J=1,NQ                                                          00014560
      IF(I-N.EQ.NPQ(J)) GO TO 20                                            00014570
81    CONTINUE                                                              00014580
      GO TO 50                                                              00014590
40    TO=TKRO(I-N)                                                          00014600
      TW=TKRW(I-N)                                                          00014610
      TG=TKRG(I-N)                                                          00014620
      RETURN                                                                00014630
      END                                                                   00014640
C                                                                           00014650
C                                                                           00014660
      SUBROUTINE OLDT(NTYPE)                                                00014670
C  PURPOSE OF SUBROUTINE:                                                   00014680
C    TO CALCULATE THE VALUE OF AN INTERMEDIATE FINITE DIFFERENCE            00014690
C      TERM AT THE OLD TIME VALUES                                          00014700
      COMMON/FDIFF/DT,EPS,H(20),TF,THETA,ICT(250),NAP                       00014710
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW       00014720
      COMMON/PROD/QO(50),QW(50),QG(50),NPQ(50),NQ                           00014730
      COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                     00014740
      COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                     00014750
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                00014760
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                  00014770
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                              00014780
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                 00014790
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                          00014800
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP             00014810
      EQUIVALENCE (GHDL,GHDW,GHDG)                                          00014820
      DATA CTX,CTY,GTX,GTY/4*0.0/                                           00014830
      WRITE(IO,1004)                                                        00014840
1004  FORMAT('1CELL #   PWF',5X,'QO',7X,'QG',7X,'QW',7X,'INJ.OR PROD')     00014850
```

```
C  TEMPORARY; FOR SAMPLE PROBLEM WITH MANY TIME STEPS ONLY*************00014860
      IF(DT.LE.0.0) GO TO 12                                            00014870
      PWF=1500.0                                                        00014880
      AKQ=1.0                                                           00014890
      S=0.0                                                             00014900
      I=1                                                               00014910
      NIND=0                                                            00014920
      CALL SORSNK(AKQ,PWF,S,I,NIND)                                     00014930
      GO TO 13                                                          00014940
C  END OF TEMPORARY*****************************************************00014950
12    READ(IN,1001) NQ,DT                                               00014960
1001  FORMAT(I4,F10.0)                                                  00014970
      DO 11 I=1,NQ                                                      00014980
      READ(IN,1002) NIND,NPQ(I),QO(I),QG(I),QW(I),PWF,S,AKQ             00014990
1002  FORMAT(2I4,6F10.0)                                                00015000
      CALL SORSNK(AKQ,PWF,S,I,NIND)                                     00015010
11    CONTINUE                                                          00015020
13    KQ=1                                                              00015030
      J=1                                                               00015040
      DO 10 I=IFRST,ILAST                                               00015050
      IF(ICN(J).NE.I) GO TO 30                                          00015060
      J=J+1                                                             00015070
      OT(I)=0.0                                                         00015080
      GO TO 10                                                          00015090
30    CALL BOUND(I,PIP1,PIM1,PIPN,PIMN,PO)                              00015100
      TLX=(TLAMOX(I)+TLAMWX(I)+TLAMGX(I))*(PIP1-PO(I))                  00015110
      TLY=(TLAMOY(I)+TLAMWY(I)+TLAMGY(I))*(PIPN-PO(I))                  00015120
      IF(I.LE.NETA) GO TO 41                                            00015130
      SLAMOY=TLAMOY(I-NETA)                                             00015140
      SLAMGY=TLAMGY(I-NETA)                                             00015150
      SLAMWX=TLAMWX(I-NETA)                                             00015160
42    SLAMOX=TLAMOX(I-1)                                                00015170
      SLAMGX=TLAMGX(I-1)                                                00015180
      SLAMWX=TLAMWX(I-1)                                                00015190
      GO TO 45                                                          00015200
41    SLAMOY=0.0                                                        00015210
      SLAMGY=0.0                                                        00015220
      SLAMWY=0.0                                                        00015230
      IF(I.NE.IFRST) GO TO 42                                           00015240
      SLAMOX=0.0                                                        00015250
      SLAMGX=0.0                                                        00015260
      SLAMWX=0.0                                                        00015270
45    SLX=(SLAMOX+SLAMGX+SLAMWX)*(PO(I)-PIM1)                           00015280
      SLY=(SLAMOY+SLAMGY+SLAMWY)*(PO(I)-PIMN)                           00015290
      IF(NTYPE) 31,20,50                                                00015300
```

```
C  CONSIDER CAPILLARY EFFECTS                                          00015310
50     CALL EBOUND(PCWIP1,PCWIM1,PCWIPN,PCWIMN,PCWO)                  00015320
       C1=TLAMWY(I)*(PCWIPN-PCWO(I))                                  00015330
       C2=SLAMWY*(PCWO(I)-PCWIMN)                                     00015340
       C5=TLAMWX(I)*(PCWIP1-PCWO(I))                                  00015350
       C6=SLAMWX*(PCWO(I)-PCWIM1)                                     00015360
       CALL EBOUND(PCGIP1,PCGIM1,PCGIPN,PCGIMN,PCGO)                  00015370
       C3=TLAMGY(I)*(PCGIPN-PCGO(I))                                  00015380
       C4=SLAMGY*(PCGO(I)-PCGIMN)                                     00015390
       C7=TLAMGX(I)*(PCGIP1-PCGO(I))                                  00015400
       C8=SLAMGX*(PCGO(I)-PCGIM1)                                     00015410
       CTX=C6-C5+C7-C8                                                00015420
       CTY=C2-C1+C3-C4                                                00015430
       IF(NTYPE.NE.2) GO TO 20                                        00015440
C  CONSIDER GRAVITY EFFECTS                                           00015450
31     CALL EBOUND(GHIP1,GHIM1,GHIPN,GHIMN,GH)                        00015460
       CALL EBOUND(DLIP1,DLIM1,DLIPN,DLIMN,DL)                        00015470
       GHDL=GH(I)*DL(I)                                               00015480
       G1=TLAMOY(I)*(GHIPN*DLIPN-GHDL)                                00015490
       G2=SLAMOY*(GHDL-GHIMN*DLIMN)                                   00015500
       G7=TLAMOX(I)*(GHIP1*DLIP1-GHDL)                                00015510
       G8=SLAMOX*(GHDL-GHIM1*DLIM1)                                   00015520
       CALL EBOUND(DWIP1,DWIM1,DWIPN,DWIMN,DW)                        00015530
       GHDW=GH(I)*DW(I)                                               00015540
       G3=TLAMWY(I)*(GHIPN*DWIPN-GHDW)                                00015550
       G4=SLAMWY*(GHDW-GHIMN*DWIMN)                                   00015560
       G9=TLAMWX(I)*(GHIP1*DWIP1-GHDW)                                00015570
       G10=SLAMWX*(GHDW-GHIM1*DWIM1)                                  00015580
       CALL EBOUND(DGIP1,DGIM1,DGIPN,DGIMN,DG)                        00015590
       GHDG=GH(I)*DG(I)                                               00015600
       G5=TLAMGY(I)*(GHIPN*DGIPN-GHDG)                                00015610
       G6=SLAMGY*(GHDG-GHIMN*DGIMN)                                   00015620
       G11=TLAMGX(I)*(GHIP1*DGIP1-GHDG)                               00015630
       G12=SLAMGX*(GHDG-GHIM1*DGIM1)                                  00015640
       GTY=G6-G5+G4-G3+G2-G1                                          00015650
       GTX=G12-G11+G10-G9+G8-G7                                       00015660
20     B2=0.0                                                         00015670
       KY=1+(I-1)/NETA                                                00015680
       F=(THETA-1.)                                                   00015690
       IF(I.NE.NPQ(KQ)) GO TO 40                                      00015700
       IF(DL(I).NE.0.0) B2=QO(KQ)/DL(I)                               00015710
       IF(DG(I).NE.0.0) B2=B2+QG(KQ)/DG(I)                            00015720
       IF(DW(I).NE.0.0) B2=B2+QW(KQ)/DW(I)                            00015730
       KQ=KQ+1                                                        00015740
       B2=B2*DT                                                       00015750
```

```
40      OT(I)=F/DY(KY)*(TLY-SLY+GTY+CTY)+F/DX(I-NETA*(KY-1))*(TLX-SLX+GTX+00015760
       &CTX)-VOL(I)*B1(I)*PO(I)/DT+B2                                     00015770
10      CONTINUE                                                          00015780
        RETURN                                                            00015790
        END                                                               00015800
C                                                                         00015810
C                                                                         00015820
        SUBROUTINE CPGRV(HTX,HTY,SLAMX,SLAMY,TLAMX,TLAMY,I,NTYPE)         00015830
C   PURPOSE OF SUBROUTINE:                                                00015840
C     TO DETERMINE THE VALUES OF THE CAPILLARY AND GRAVITY TERMS OF       00015850
C       THE FINITE DIFFERENCE EQUATION                                    00015860
        COMMON/IBMBTX/TLAMOX(500),TLAMGX(500),TLAMWX(500)                 00015870
        COMMON/IBMBTY/TLAMOY(500),TLAMGY(500),TLAMWY(500)                 00015880
        COMMON/DNSTY/DL(500),DW(500),DG(500)                              00015890
        COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                          00015900
        COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW   00015910
        COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)            00015920
        EQUIVALENCE (PCWIP1,PCGIP1,DLIP1,DWIP1,DGIP1),(PCWIM1,PCGIM1,DLIM100015930
       &,DWIM1,DGIM1),(PCWIM1,PCGIM1,DLIM1,DWIM1,DGIM1),(PCWIPN,PCGIPN,DLI00015940
       &PN,DWIPN,DGIPN),(PCWIMN,PCGIMN,DLIMN,DWIMN,DGIMN),(H7,H3),(H17,H1300015950
       &),(H9,H1),(H19,H11),(H10,H4),(H20,H14),(GHDL,GHDW,GHDG)            00015960
        DATA H2,H8,H12,H18/4*0.0/                                         00015970
        IF(HT(I).EQ.0.0) GO TO 10                                         00015980
        IF(I.LE.NETA) GO TO 41                                            00015990
        SLAMOY=TLAMOY(I-NETA)                                             00016000
        SLAMGY=TLAMGY(I-NETA)                                             00016010
        SLAMWY=TLAMWY(I-NETA)                                             00016020
42      SLAMOX=TLAMOX(I-1)                                                00016030
        SLAMGX=TLAMGX(I-1)                                                00016040
        SLAMWX=TLAMWX(I-1)                                                00016050
        GO TO 45                                                          00016060
41      SLAMOY=0.0                                                        00016070
        SLAMGY=0.0                                                        00016080
        SLAMWY=0.0                                                        00016090
        IF(I.NE.IFRST) GO TO 42                                           00016100
        SLAMOX=0.0                                                        00016110
        SLAMGX=0.0                                                        00016120
        SLAMWX=0.0                                                        00016130
45      IF(NTYPE) 70,50,90                                                00016140
C   CONSIDER CAPILLARY EFFECTS                                            00016150
90      CALL EBOUND(PCWIP1,PCWIM1,PCWIPN,PCWIMN,PCWO)                     00016160
        H7=TLAMWY(I)*(PCWIPN-PCWO(I))                                     00016170
        H8=SLAMWY*(PCWO(I)-PCWIMN)                                        00016180
        H17=TLAMWX(I)*(PCWIP1-PCWO(I))                                    00016190
        H18=SLAMWX*(PCWO(I)-PCWIM1)                                       00016200
```

```
      CALL EBOUND(PCGIP1,PCGIM1,PCGIPN,PCGIMN,PCGO)                    00016210
      H9=TLAMGY(I)*(PCGIPN-PCGO(I))                                    00016220
      H10=SLAMGY*(PCWO(I)-PCGIMN)                                      00016230
      H19=TLAMGX(I)*(PCGIP1-PCGO(I))                                   00016240
      H20=SLAMGX*(PCGO(I)-PCGIM1)                                      00016250
      H8=H8-H7+H9-H10                                                  00016260
      H18=H18-H17+H19-H20                                              00016270
      IF(NTYPE.NE.2) GO TO 50                                          00016280
C  CONSIDER GRAVITY EFFECTS                                            00016290
70    CALL EBOUND(GHIP1,GHIM1,GHIPN,GHIMN,GH)                          00016300
      CALL EBOUND(DLIP1,DLIM1,DLIPN,DLIMN,DL)                          00016310
      GHDL=GH(I)*DL(I)                                                 00016320
      H1=TLAMOY(I)*(GHIPN*DLIPN-GHDL)                                  00016330
      H2=SLAMOY*(GHDL-GHIMN*DLIMN)                                     00016340
      H11=TLAMOX(I)*(GHIP1*DLIP1-GHDL)                                 00016350
      H12=SLAMOX*(GHDL-GHIM1*DLIM1)                                    00016360
      CALL EBOUND(DWIP1,DWIM1,DWIPN,DWIMN,DW)                          00016370
      GHDW=GH(I)*DW(I)                                                 00016380
      H3=TLAMWY(I)*(GHIPN*DWIPN-GHDW)                                  00016390
      H4=SLAMWY*(GHDW-GHIMN*DWIMN)                                     00016400
      H13=TLAMWX(I)*(GHIP1*DWIP1-GHDW)                                 00016410
      H14=SLAMWX*(GHDW-GHIM1*DWIM1)                                    00016420
      CALL EBOUND(DGIP1,DGIM1,DGIPN,DGIMN,DG)                          00016430
      GHDG=GH(I)*DG(I)                                                 00016440
      H5=TLAMGY(I)*(GHIPN*DGIPN-GHDG)                                  00016450
      H6=SLAMGY*(GHDG-GHIMN*DGIMN)                                     00016460
      H15=TLAMGX(I)*(GHIP1*DGIP1-GHDG)                                 00016470
      H16=SLAMGX*(GHDG-GHIM1*DGIM1)                                    00016480
      H2=H2-H3-H1+H4-H5+H6                                             00016490
      H12=H12-H13-H11+H14-H15+H16                                      00016500
50    HTX=H12+H18                                                      00016510
      HTY=H2+H8                                                        00016520
      TLAMX=TLAMOX(I)+TLAMGX(I)+TLAMWX(I)                              00016530
      SLAMX=SLAMOX+SLAMGX+SLAMWX                                       00016540
      TLAMY=TLAMOY(I)+TLAMGY(I)+TLAMWY(I)                              00016550
      SLAMY=SLAMOY+SLAMGY+SLAMWY                                       00016560
      RETURN                                                           00016570
10    HTX=0.0                                                          00016580
      HTY=0.0                                                          00016590
      SLAMX=0.0                                                        00016600
      SLAMY=0.0                                                        00016610
      TLAMX=0.0                                                        00016620
      TLAMY=0.0                                                        00016630
      RETURN                                                           00016640
      END                                                              00016650
```

```
C                                                                               00016660
C                                                                               00016670
      SUBROUTINE UPDATE                                                         00016680
C   PURPOSE OF SUBROUTINE:                                                      00016690
C     TO UPDATE ALL VALUES AT THE NEW PRESSURE                                  00016700
      COMMON/SATN/SW(500),SG(500),SO(500),CF                                    00016710
      COMMON/CPLRY/PCWO(500),PCGO(500),B1(500)                                  00016720
      COMMON/PCAPO/APCO(10),PPCO(10),NPCO,IFPCO                                 00016730
      COMMON/PCAPG/APCG(10),PPCG(10),NPCG,IFPCG                                 00016740
      COMMON/ROCK/PHI(500),HT(500),ZEL(500),VOL(500),GH(500)                    00016750
      COMMON/DNSTY/DL(500),DW(500),DG(500)                                      00016760
      COMMON/COMP/T,XL(500),X(5000),Y(5000),N,NC(50)                            00016770
      COMMON/TRIDAG/A(500),B(500),C(500),D(500),IFRST,ILAST                     00016780
      COMMON/WTRV/AUW(10),PUW(10),NUW,IFUW                                      00016790
      COMMON/INOUT/IN,IO,NFIP,NLMDA,NSAT,NPOT,NCOMP,NTKH,NPLKUP                 00016800
      COMMON/SYSTEM/DX(20),DY(20),ICN(30),METHOD,NIC,NCELLS,NETA,NROW           00016810
      COMMON/SAVE/OT(500),PN(500),PNP(500),PO(500)                              00016820
      COMMON/COMP1/ACO(10),ACW(10),ACG(10),PCO(10),PCW(10),PCG(10)              00016830
      COMMON/COMP1/NCO,NCW,NCG,IFCO,IFCW,IFCG                                   00016840
      DIMENSION POTL(500),POTW(500),POTG(500),Z(5000)                           00016850
      DATA N1,N2,N3/'OIL ','GAS ','H2O '/                                       00016860
      EQUIVALENCE (X(1),Z(1)),(POTL(1),A(1)),(POTW(1),B(1)),(POTG(1),C(100016870
     &))                                                                        00016880
      DO 10 I=IFRST,ILAST                                                       00016890
      POTL(I)=PNP(I)-DL(I)*GH(I)                                                00016900
      POTG(I)=PNP(I)+PCGO(I)-DG(I)*GH(I)                                        00016910
10    POTW(I)=PNP(I)+PCWO(I)-DW(I)*GH(I)                                        00016920
      IF(NPOT.EQ.0) GO TO 41                                                    00016930
      WRITE(IO,1001)                                                            00016940
1001  FORMAT('1POTENTIAL MATRICES')                                             00016950
      DO 42 I=NETA,NCELLS,NETA                                                  00016960
      WRITE(IO,1002) N1,(POTL(I+J-NETA),J=1,NETA)                               00016970
1002  FORMAT(//,1X,A4,10F10.3)                                                  00016980
      WRITE(IO,1003) N2,(POTG(I+J-NETA),J=1,NETA)                               00016990
1003  FORMAT(1X,A4,10F10.3)                                                     00017000
      WRITE(IO,1003) N3,(POTW(I+J-NETA),J=1,NETA)                               00017010
42    CONTINUE                                                                  00017020
41    J=1                                                                       00017030
      DO 11 I=IFRST,ILAST                                                       00017040
      IF(ICN(J).NE.I) GO TO 12                                                  00017050
      J=J+1                                                                     00017060
      GO TO 11                                                                  00017070
12    CALL BOUND(I,POTLP1,POTLM1,POTLPN,POTLMN,POTL)                            00017080
      CALL CPGRV(HTX,HTY,SLAMX,SLAMY,TLAMX,TLAMY,I,0)                           00017090
      C1=DL(I)*(TLAMX*(POTLP1-POTL(I))-SLAMX*(POTL(I)-POTLM1)+TLAMY*(POT00017100
```

```
     &LPN-POTL(I))-SLAMY*(POTL(I)-POTLMN))                                          00017110
      CALL EBOUND(POTGP1,POTGM1,POTGPN,POTGMN,POTG)                                 00017120
      C2=DG(I)*(TLAMX*(POTGP1-POTG(I))-SLAMX*(POTG(I)-POTGM1)+TLAMY*(POT00017130
     &GPN-POTG(I))-SLAMY*(POTG(I)-POTGMN))                                          00017140
      DTT=DT/VOL(I)                                                                 00017150
      C1=DTT*C1                                                                     00017160
      C2=DTT*C2                                                                     00017170
      ST=SO(I)*DL(I)+SG(I)*DG(I)                                                    00017180
      L=N*(I-1)                                                                     00017190
      DO 20 K=1,N                                                                   00017200
20    Z(L+K)=ST*(XL(I)*(X(L+K)-Y(L+K))+Y(L+K))+C1*X(L+K)+C2*Y(L+K)                  00017210
11    CONTINUE                                                                      00017220
      CALL SATRN                                                                    00017230
      J=1                                                                           00017240
      DO 30 I=IFRST,ILAST                                                           00017250
      IF(ICN(J).NE.I) GO TO 32                                                      00017260
      J=J+1                                                                         00017270
      GO TO 30                                                                      00017280
32    L=N*(I-I)                                                                     00017290
      ST=SO(I)*DL(I)+SG(I)*DG(I)                                                    00017300
      DO 31 K=1,N                                                                   00017310
31    Z(L+K)=Z(L+K)/ST                                                              00017320
      CALL FLASH(PNP(I),L,I)                                                        00017330
      CALL CRTPRP(PNP(I),L,I)                                                       00017340
      CALL LOOKUP(PCWO(I),SO(I),APCO,PPCO,NPCO,IFPCO)                               00017350
      CALL LOOKUP(PCGO(I),SG(I),AFCG,PPCG,NPCG,IFPCG)                               00017360
      CALL LOOKUP(C1,PNP(I),ACO,PCO,NCO,IFCO)                                       00017370
      CALL LOOKUP(C2,PNP(I),ACW,PCW,NCW,IFCW)                                       00017380
      CALL LOOKUP(C3,PNP(I),ACG,PCG,NCG,IFCG)                                       00017390
      B1(I)=CF+SO(I)*C1+SW(I)*C2+SG(I)*C3                                           00017400
30    CONTINUE                                                                      00017410
      IF(NCOMP.EQ.0) RETURN                                                         00017420
      WRITE(IO,1006)                                                                00017430
1006  FORMAT('1OVERALL COMPOSITION MATRIX')                                         00017440
      DO 51 I=NETA,NCELLS,NETA                                                      00017450
      WRITE(IO,1007) (NC(J),J=1,5),(Z(N*(I+J-NETA)-9),J=1,NETA)                     00017460
1007  FORMAT(//,1X,5A2,10F10.6)                                                     00017470
      DO 51 K=2,N                                                                   00017480
      NA=(K-1)*5                                                                    00017490
      WRITE(IO,1008) (NC(J+NA),J=1,5),(Z(N*(I+J-NETA-1)+K),J=1,NETA)                00017500
1008  FORMAT(1X,5A2,10F10.6)                                                        00017510
51    CONTINUE                                                                      00017520
      RETURN                                                                        00017530
      END                                                                           00017540
```

FOOTNOTES


1. Morris Muskat, Physical Principles of Oil Production, (New York: McGraw-Hill Book Co., 1949), p. 381.

2. F. B. Canfield, "Estimate K-Values With the Computer," Hydrocarbon Processing 50 (April 1971):137.

3. D. D. Fussell and L. Yarborough, "The Effect of Phase Data on Liquid Recovery During Cycling of a Gas Condensate Reservoir." Paper presented at the 68'th National Meeting of AIChE, Houston, Texas, 28 February, 1971.

4. J. M. Smith and H. C. Van Ness, Introduction to Chemical Engineering Thermodynamics (New York: McGraw-Hill Book Co., 1959).

5. O. Hougen, K. Watson, and R. Ragatz, Chemical Process Principles--Part II (New York: John Wiley & Sons, Inc., 1966).

6. R. Simon and L. Yarborough, "A Critical Pressure Correlation for Gas-Solvent-Reservoir Oil Systems," Trans. AIME 228 (1963): 556.

7. D. O. Etter and W. B. Kay, "Critical Properties of Mixtures of Normal Paraffin Hydrocarbons," J. Chem. Eng. Data 6 (1961): 409.

8. R. C. MacDonald, "Reservoir Simulation with Interphase Mass Transfer (Ph.D. dissertation, University of Texas at Austin, 1972).

9. L. C. Yen and S. S. Woods, "A Generalized Equation for Computer Calculation of Liquid Densities," AIChE J., 12 (January 1966): 95-99.

10. E. T. Guerrero, "Reservoir Engineering 87. How to Find Gas and Liquid Densities from Phase-Behavior Data," Oil & Gas J. 64 (March 1966): 98-104.

11. H. T. Kennedy, C. H. Bowman, A. N. Crownover, and E. P. Miesch, "Recent Correlation of Hydrocarbon Properties

with Composition," J. Pet. Tech. 17 (September 1965): 1105-1110.

12. S. M. Avasthi and H. T. Kennedy, "The Prediction of Volumes, Compressibilities, and Thermal Expansion Coefficients of Hydrocarbon Mixtures," Soc. Pet. Eng. J. 8 (June 1968): 95-106.

13. G. H. Alani and H. T. Kennedy, "Volumes of Liquid Hydrocarbons at High Temperatures and Pressures," Trans. AIME 219 (1960): 288-292.

14. Samuel M. Selby, ed., CRC Standard Mathematical Tables, 15th ed. (Cleveland, Ohio: The Chemical Rubber Co., 1967).

15. H. H. Reamer, B. H. Sage, and W. N. Lacey, "Volumetric Behavior of Hydrogen Sulfide," Ind. Eng. Chem. 42 (1950): 140.

16. A. Van Itterbeek and O. Verbeke, "Density of Liquid Nitrogen and Argon as a Function of Pressure and Temperature," Physica 26 (1960): 931.

17. D. M. Newitt, M. U. Pai, N. R. Kuloor, and J. A. W. Huggil, "Carbon Dioxide," Therm. Func. Gases 1 (1962): 102.

18. Byung-Ik Lee and W. C. Edmister, "New Three-Parameter Equation of State," Ind. & Eng. Chem. Fund. 10 (February 1971): 32-35.

19. J. S. Nolen, "Numerical Simulation of Compositional Phenomean in Petroleum Reservoirs," paper presented at the 3rd SPE Symposium on Numerical Simulation of Reservoir Performance, Houston, Texas, 10 January 1973.

20. M. B. Standing and D. L. Katz, "Density of Natural Gases," Trans. AIME 146 (1942): 140-149.

21. W. B. Kay, "Density of Hydrocarbon Gases and Vapor," Ind. & Eng. Chem. 28 (1936): 1014-1019.

22. A. M. Sarem, "Z-Factor Equation Developed for Use in Digital Computers," Oil & Gas J. 59 (September 1961): 118.

23. E. L. Heric, "On the Viscosity of Ternary Mixtures," J. Chem. & Eng. Data 11 (January (1966): 66-68.

24. N. L. Carr, R. Kobayashi, and D. B. Burrows, "Viscosity of Hydrocarbon Gases Under Pressure," Trans. AIME 201 (1954): 264-272.

25. J. R. Dempsey, "Computer Routine Treats Gas Viscosity as a Variable," Oil & Gas J. 63 (August 1965): 141-143.

26. A. L. Lee, K. E. Starling, J. P. Dolan, and R. T. Ellington, "Viscosity Correlation for Light Hydrocarbon Systems," AIChE J. 10 (September 1964): 694-697.

27. J. E. Little and H. T. Kennedy, "A Correlation of the Viscosity of Hydrocarbon Systems with Pressure, Temperature and Composition," Soc. Pet. Eng. J. 8 (June 1968): 157-162.

28. J. Lohrenz, B. G. Bray, and C. R. Clark, "Calculating Viscosities of Reservoir Fluids from Their Compositions," J. Pet. Tech. 16 (1964): 1171-1176.

29. A. L. Lee, M. H. Gonzalez, and B. E. Eakin, "The Viscosity of Natural Gases," J. Pet. Tech. 18 (1966): 997-1000.

30. R. C. Reid and T. K. Sherwood, The Properties of Gases and Liquids, 2nd ed. (New York: McGraw-Hill Book Co., 1966).

31. Henry B. Crichlow, Petroleum Reservoir Simulation (Norman, Oklahoma: Petroleum Training and Technical Services, 1974), pp. 217-220.

32. G. Birkoff, R. S. Varga, and D. Yound, "Alternating Direction Implicit Methods," Advances in Computers 3 (1962): 189-273.

33. H. L. Stone, "Probability Model for Estimating Three-Phase Relative Permeability," J. Pet. Tech. 22 (February 1970): 214-218.

# BIBLIOGRAPHY


Alani, G. H. and Kennedy, H. T. "Volumes of Liquid Hydrocarbons at High Temperatures and Pressures." Trans. AIME 219 (1960): 288-292.

Avasthi, S. M. and Kennedy, H. T. "The Prediction of Volumes, Compressibilities, and Thermal Expansion Coefficients of Hydrocarbon Mixtures." Soc. Pet. Eng. J. 8 (June 1968): 95-106.

Birkoff, G., Varga, R. S., Yound, D. "Alternating Direction Implicit Methods." Advances in Computers 3 (1962): 189-273.

Bjordammen, J. and Coats, K. H. "Comparison of Alternating-Direction and Successive Overrelaxation Techniques in Simulation of Reservoir Fluid Flow." Soc. Pet. Eng. J. 9 (March 1969): 47-58.

Blair, P. M. and Weinaug, C. F. "Solution of Two-Phase Flow Problems Using Implicit Difference Equations." Soc. Pet. Eng. J. 9 (December 1969): 417-424.

Briggs, J. E. and Dixon, T. N. "Some Practical Considerations in the Numerical Solution of Two-Dimensional Reservoir Problems." Soc. Pet. Eng. J. 8 (June 1968): 185-194.

Brinkman, F. H. and Sicking, J. N. "Equilibrium Ratios for Reservoir Studies." Trans. AIME 219 (1960): 313-319.

Canfield, F. B. "Estimate K-Values with the Computer." Hydrocarbon Processing 50 (April 1971): 137-138.

Carr, N. L., Kobayashi, R., and Burrows, D. B. "Viscosity of Hydrocarbon Gases Under Pressure." Trans. AIME 201 (1954): 264-272.

Chao, K. C. and Seader, J. D. "A General Correlation of Vapor-Liquid Equilibria in Hydrocarbon Mixtures." AIChE J. 7 (December 1961): 598-605.

Chappelear, J. E. and Hirasaki, G. J. "A Model of Oil-Water Coning for 2-D Areal Reservoir Simulation." Paper presented at the 49th Annual Fall Meeting of the Soceity of Petroleum Engineers, Houston, Texas, 6 October 1974.

Chu, C. and Trimble, A. E. "Numerical Simulation of Steam Displacement--Field Performance Applications." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Coats, K. H. "Use and Misuse of Reservoir Simulation Models." J. Pet. Tech. 21 (November 1969): 1391-1398.

Cook, R. E., Jacoby, R. H., Ramesh, A. B. "A Beta-Type Reservoir Simulator for Approximating Compositional Effects During Gas Injection." Paper presented at the 3rd SPE Symposium on Numerical Simulation of Reservoir Performance, Houston, Texas, 10 January 1973.

Craft, B. C. and Hawkins, M. F. Applied Petroleum Reservoir Engineering. Englewood Cliffs, N.J.: Prentice-Hall, 1959.

Craig, F. F., J. "Effect of Reservoir Description on Performance Predictions." J. Pet. Tech. 22 (October 1970): 1239-1245.

Crichlow, Henry B. Petroleum Reservoir Simulation. Norman, Okla.: Petroleum Training and Technical Services, 1974.

Dempsey, J. R. "Computer Routine Treats Gas Viscosity as a Variable." Oil & Gas J. 63 (August 1965): 141-143.

Engineering Data Book. Tulsa, Okla.: Natural Gas Processors Suppliers Association, 1972.

Etter, D. O. and Kay, W. B. "Critical Properties of Mixtures of Normal Paraffin Hydrocarbons." J. Chem. Eng. Data 6 (1961): 409.

Fagin, R. G. and Stewart, C. H., Jr. "A New Approach to the Two-Dimensional Multiphase Reservoir Simulator." Soc. Pet. Eng. J. 6 (June 1966): 175-182.

Finol, A. and Farouq Ali, S. M. "Numerical Simulation of Oil Production with Simultaneous Ground Subsidence." Paper presented at the SPE-European Spring Meeting of the Society of Petroleum Engineers, Amsterdam, The Netherlands, 29 May 1974.

Frick, Thomas, C., ed. Petroleum Production Handbook, Dallas, Texas: The Society of Petroleum Engineers of AIME, 1962.

Fussell, D. D. and Yarborough, L. "The Effect of Phase Data on Liquid Recovery During Cycling of a Gas Condensate Reservoir." Paper presented at the 68th National Meeting of AIChE, Houston, Texas, 28 February 1971.

Guerrero, E. T. "Reservoir Engineering 87. How to Find Gas and Liquid Densities from Phase-Behavior Data." Oil & Gas J. 64 (March 1966): 98-104.

Hala, E., Pick, J., Friad, V., and Vilim, O. Vapour-Liquid Equilibirum. New York: Pergamon Press, 1967.

Harris, D. G. "The Role of Geology in Reservoir Simulation Studies." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Heric, E. L. "On the Viscosity of Ternary Mixtures." J. Chem. & Eng. Data 11 (January 1966): 66-68.

Herning, F. and Zipperer, L. "Calculation of the Viscosity of Technical Gas Mixtures from the Viscosity of Individual Gases." Das Gas-Und Wasserfach 79 (1936): 49-73.

Hirasaki, G. J. and O'Dell, P. M. "Representation of Reservoir Geometry for Numerical Simulation." Soc. Pet. Eng. J. 22 (December 1970): 393-404.

Hougen, O.; Watson, K; and Ragatz, R. Chemical Process Principles--Part II. New York: John Wiley & Sons, Inc., 1966.

Houpert, A. H. and Thelliez, M. B. "Predicting the Viscosity of Hydrocarbon Liquid Phases from Their Composition." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Jacks, H. H., Smith, O. J. E., and Maddox, C. C. "The Modeling of a Three-Dimensional Reservoir with a Two-Dimensional Reservoir Simulator--The Use of Dynamic Pseudo-Function." Soc. Pet. Eng. J. 13 (June 1973): 175-185.

Jones, D. M. and Erbar, J. "Computer Determination of Data-Matched Equilibrium Ratios." J. Pet. Tech. 22 (August 1970): 1007-1011.

Kay, W. B. "Density of Hydrocarbon Gases and Vapor." Ind. & Eng. Chem. 28 (1936): 1014-1019.

Kennedy, H. T., Bowman, C. H., Crownover, A. N. and Miesch, E. P. "Recent Correlations of Hydrocarbon Properties with Composition." J. Pet. Tech. 17 (September 1965): 1105-1110.

Lee, A. L., Gonzelez, M. H., and Eakin, B. E. "The Viscosity of Natural Gases." J. Pet. Tech. 18 (1966): 997-1000.

Lee, A. L., Starling, K. E., Dolan, J. P., and Ellington, R. T. "Viscosity Correlation for Light Hydrocarbon Systems." AIChE J. 10 (September 1964): 694-697.

Lee, Byung-Ik and Edmister, W. C. "New Three-Parameter Equation of State." Ind. & Eng. Chem. Fund. 10 (February 1971): 32-35.

Little, J. E. and Kennedy, H. T. "A Correlation of the Viscosity of Hydrocarbon Systems with Pressure, Temperature and Composition." Soc. Pet. Eng. J. 8 (June 1968): 157-162.

Lohrenz, J., Bray, B. G., and Clark, C. R. "Calculating Viscosities of Reservoir Fluids from their Compositions." J. Pet. Tech. 16 (1964): 1171-1176.

MacDonald, R. C. "Reservoir Simulation with Interphase Mass Transfer." Ph.D. dissertation, University of Texas at Austin, 1972.

McCain, W. D., Jr. The Properties of Petroleum Fluids. Tulsa, Okla.: Petroleum Publishing Co., 1973.

Mercer, J. W. and Faust, C. R. "Simulation of Water- and Vapor-Dominated Hydrothermal Reservoirs." Paper presented at the 50th Annual Fall Meeting of the Society of Petroleum Engineers, Dallas, Texas, 28 September 1975.

Muskat, Morris. The Flow of Homogeneous Fluids Through Porous Media. New York: McGraw-Hill Book Co., 1937.

Muskat, Morris. Physical Principles of Oil Production. New York: McGraw-Hill, 1949.

Newitt, D. M., Pai, M. U., Kuloor, N. R., and Huggil, J. A. W. "Carbon Dioxide." Therm. Func. Gases 1 (1962): 102.

Nolen, J. S. "Numerical Simulation of Compositional Phenomena in Petroleum Reservoirs." Paper presented at the 3rd SPE Symposium on Numerical Simulation of Reservoir Performance, Houston, Texas, 10 January 1973.

Perry, J. H. and Herron, E. H., Jr. "Three-Phase Reservoir Simulation." J. Pet. Tech. 21 (February 1969): 211-220.

Prausnitz, J. M. Molecular Thermodynamics of Fluid-Phase Equilibria. New York: Prentice-Hall, Inc., 1969.

Price, H. S. and Coats, K. H. "Direct Methods in Reservoir Simulation." Paper presented at the 3rd SPE Symposium on Numerical Simulation of Reservoir Performance, Houston, Texas, 11 January 1973.

Price, H. S. and Donohue, D. A. T. "Isothermal Displacement Processes with Interphase Mass Transfer." Soc. Pet. Eng. J. 7 (June 1967): 205-220.

Reamer, H. H., Sage, B. H., and Lacey, W. N. "Volumetric Behavior of Hydrogen Sulfide." Ind. Eng. Chem. 42 (1950): 140.

Reid, R. C. and Sherwood, T. K. The Properties of Gases and Liquids. 2nd ed. New York: McGraw-Hill Book Co., 1966.

Roebuck, I. F., Jr., Henderson, G. E., Douglas, J., Jr., and Ford, W. T. "The Compositional Reservoir Simulator: Case I--The Linear Model." Soc. Pet. Eng. J. 9 (March 1969): 115-130.

Rowe, A. M. "The Critical Composition Method--A New Convergence Pressure Method." Soc. Pet. Eng. J. 7 (March 1967): 54-60.

Sanyal, S. K. "Effect of Temperature on Petrophysical Properties of Reservoir Rocks." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Sarem, A. M. "Z-Factor Equation Developed for Use in Digital Computers." Oil & Gas J. 59 (September 1961): 118.

Selby, Samuel M., ed. CRC Standard Mathematical Tables. Cleveland, Ohio: The Chemical Rubber Co., 1967.

Seth, M. S. "A Semi-Implicit Method for Simulating Reservoir Behavior." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Settari, A. and Aziz, K. "Use of Irregular Grid in Reservoir Simulation." Soc. Pet. Eng. J. 12 (April 1972): 103-114.

Sheffield, M. "Three-Phase Fluid Flow Including Gravitational Viscous and Capillary Forces." Soc. Pet. Eng. J. 9 (June 1969): 255-269.

Shutler, N. D. "Numerical Three-Phase Model of the Two-Dimensional Steamflood Process." Soc. Pet. Eng. J. 10 (December 1970): 405-417.

Simon, R. and Yarborough, L. "A Critical Pressure Correlation for Gas-Solvent-Reservoir Oil Systems." Trans. AIME 228 (1963): 556.

Sincovec, R. F. "Numerical Reservoir Simulation Using an Ordinary Differential Equations Integrator." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Smith, J. M. and Van Ness, H. C. Introduction to Chemical Engineering Thermodynamics. New York: McGraw-Hill Book Co., 1959.

Snyder, L. J. "Two-Phase Reservoir Flow Calculations." Soc. Pet. Eng. J. 9 (June 1969): 170-182.

Spivak, A. "Mathematical Simulation of Condensate-Gas Reservoirs." Ph.D. dissertation, University of Texas at Austin, 1971.

Staggs, H. M. and Herbeck, E. G. "Reservoir Simulation Models--An Engineering Overview." J. Pet. Tech. 23 (December 1971): 1428-1436.

Standing, M. B. and Katz, D. L. "Density of Natural Gases." Trans. AIME 146 (1942): 140-149.

Starling, K. E. and Han, M. S. "Thermo Data Refined for LPG: Part 15: Industrial Applications." Hydrocarbon Processing 51 (June 1972): 107-117.

Stone, H. L. "Probability Model for Estimating Three-Phase Relative Permeability." J. Pet. Tech. 22 (February 1970): 214-218.

Thompson, F. R. and Thachuk, A. R. "Compositional Simulation of a Gas Cycling Project, Bonnie Glen D-3A Pool, Alberta, Canada." Paper presented at the Third SPE Symposium on Numerical Simulation of Reservoir Performance, Houston, Texas, 10 January 1973.

Todd, M. R., O'Dell, P. M., and Hirasaki, G. J. "Methods for Increased Accuracy in Numerical Reservoir Simulators." Soc. Pet. Eng. J. 12 (December 1972): 515-530.

Toronyi, R. M. and Farouq Ali, S. M. "Two-Phase, Two-Dimensional Simulation of a Geothermal Reservoir and the Wellbore System." Paper presented at the 49th Annual Fall Meeting of the Society of Petroleum Engineers, Houston, Texas, 6 October 1974.

Tsutsumi, G. "Mathematical Simulation of Two-Phase Flow with Interphase Mass Transfer in Petroleum Reservoirs." Ph.D. dissertation, University of Texas at Austin, 1973.

Van Itterbeek, A. and Verbeke, O. "Density of Liquid Nitrogen and Argon as a Function of Pressure and Temperature." Physica 26 (1960): 931.

Weinstein, H. G., Stone, H. L., and Kwan, T. V. "Simultaneous Solution of Multi-Phase Reservoir Flow Equations." Soc. Pet. Eng. J. 10 (June 1970): 99-110.

Yen, L. C. and Woods, S. S. "A Generalized Equation for Computer Calculation of Liquid Densities." AIChE J. 12 (January 1966): 95-99.